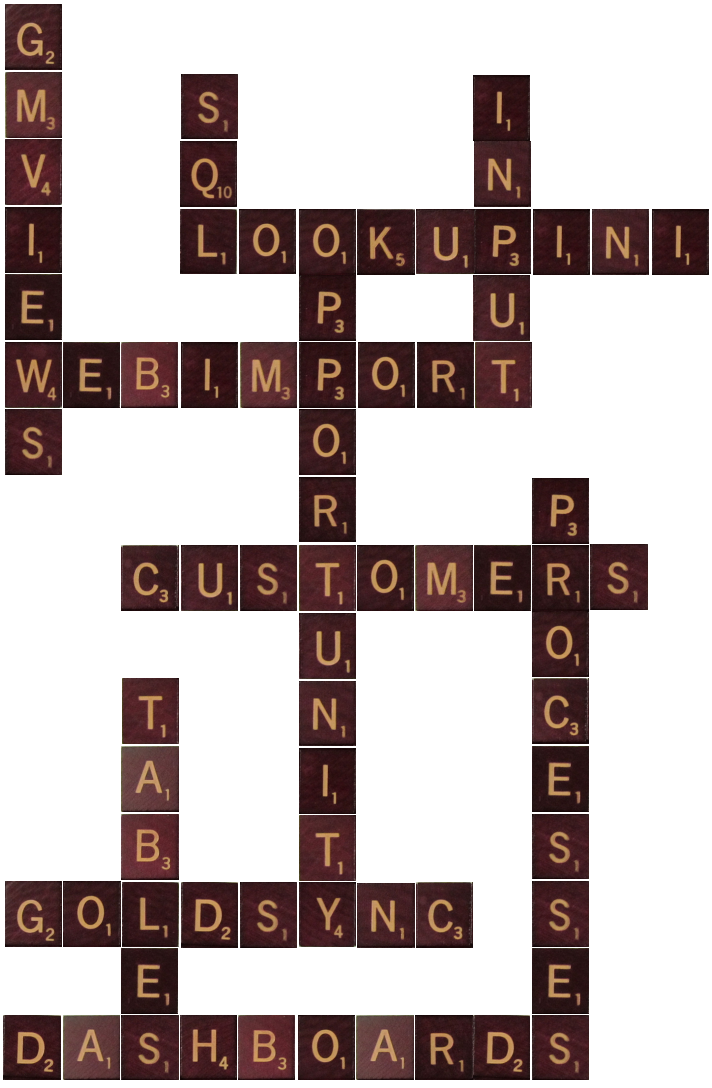
Author:
DJ Hunt

In This Book

Top GoldMine Partner 2011



GoldMine Premium - The Definitive Guide

GoldMine
Creating Customers for Life®

150 Pratt Road
Fitchburg, MA 01420

Voice: (978)342-3333
E-Mail: DJ@DJHunt.US
Website: www.DJHunt.US

GoldMine Premium - The Definitive Guide

First Printing January 2011

Trademark Acknowledgements

The Author

GoldMine for Window, 1992
Getting the Gold out of GoldMine, 1995
Sales Force Automation with GoldMine 4.0, 1999
The Hackers Guide to GoldMine, 2003
The Hacker's Guide to GoldMine 6.5, 2004
The Hacker's Guide to GoldMine 6.6, 2004
The Hacker's Guide to GoldMine 6.7, 2004
The Hacker's Guide to GoldMine 7.0 - A New Beginning, 2006
The Hacker's Guide to GoldMine Premium, 2008
GoldMine Premium - The Definitive Guide, 2011

Many GoldMine Partners are using these books as their GoldMine bibles for Technical Consultation. Most readers will have found my free advice in the forums to be helpful, or you may have used some of my 300 or so GoldMine add-on products. Beyond Gold had become a very popular free utility add-on for GoldMine in the beginning, though I was first recognized for my GoldQuote add-on. I am currently working for Computerese Inc, and working there as an GoldMine Consultant doing GoldMine Support worldwide. Not only do I speak the GoldMine rhetoric, but I can do the GoldMine dance as well.

Alas, **GoldMine Premium - The Definitive Guide** is to be my 10th and final book as I have plans to retire to become more of a part time GoldMine contributor by April 2012. I thank everyone for their kind words and accolades for my various books over the years, and I hope that you will enjoy this one just as much.

Ad Hoc Editors & Contributing Authors

I would like to thank all of those who have contributed to this book through their suggestions, and actual editing of the content of the book. Without whose assistance this book would not be going out to you as accurate, and clean as it now stands.

Special appreciation goes out to:

Carol Hunt (aka the Spouse)
Iain Wicks (Chapter 1 - Contributing Videographer)
Andrea Dominquez (Chapter 12 - Contributing Author)

Table of Contents

1

In This Chapter

Note

*Mr. Iain Wicks has agreed to permit me, albeit for a fee, to distribute his 10 Dashboard Videos along with this book. If you think that these are helpful to your needs, you may purchase the entire collection of videos at a reduced cost of **$339.15** US from:*

http://www.djhunt.us/DWSite/Docs/GoldMine/Video_GMPE.html

Dashboards really say it all as far as the new tool on the block in GoldMine Premium. I can't tell you how excited and then disappointed we were to see this module in GoldMine Premium. Excited because everyone has been waiting for this tool for such a long time. Disappointed because there is no documentation on Dashboards at all. My partner James McCracken may be writing a book on the Dashboards in the future, but there is very little to date.

Hence this, the Dashboards chapter, replaces the former Chapter 1 that was published in The Hacker's Guide series of books. I will do the best that I can to explain the basics of Dashboards, however, it would take an entire book to do Dashboards justice, and I simply don't have room in this book.

Dashboards are a graphical way of displaying your GoldMine Premium data, as opposed to, let's say, GoldMine Reports. Accessing the Dashboards is very simple. One can simply click on **Dashboards** on the Outlook style toolbar to the left of the GoldMine Premium screen or one could utilize the GoldMine Premium menu:

Go To
Dashboards

Either way will bring forth the **Dashboards** dialog form as shown below in Figure 1-1:

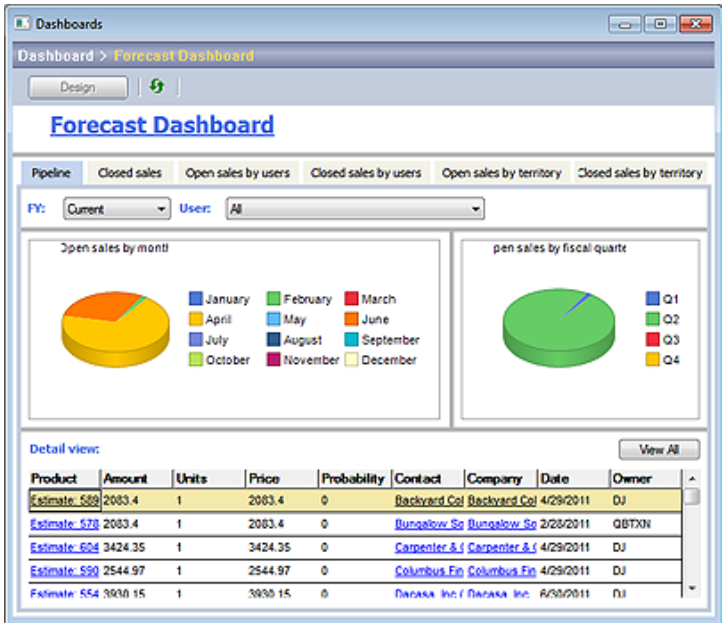


Figure 1-1

Please excuse any distortion in this image. So as to fit this image within the constraints of this page, there had to be a large amount of image compression.

This particular screen shot is that of the **Forecast Dashboard** with its 6 tab that cover **Pipeline**, **Closed sales**, **Open sales by users**, **Closed sales by users**, **Open sales by territory**, and lastly **Close sales by territory**.

While the Dashboards are open let's take a look at the Quick Toolbar to the left in GoldMine Premium as it should not contain all of your installed Dashboards. You may see what mine looks like on the next page in Figure 1-2, and yours should look similar to this. In the default state, one has 2 categories, **Management** containing 7 dashboards, and **Sales** containing 3 dashboards.

Tip

*The **Activity Aging Random Example** is not one of the GoldMine Premium default dashboards as it is one that I Imported as a test of the Dashboard Import feature during the Beta cycle.*

*Additionally, the **Undefined** category shown in Figure 1-2 was developed as part of my testing.*

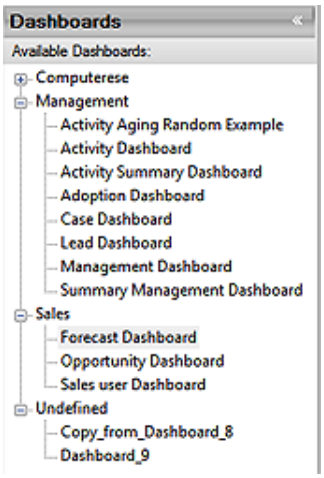


Figure 1-2

A right-click over any of the dashboards and you will have the ability to:

New
Edit
Copy
Properties
Delete
Import
Export

Notice that there are no buttons with which to do these activities within the Dashboards dialog form. There is one hotlink at the bottom of the Quick Toolbar that will allow one to create a **New Dashboard...**, but beyond that your only way to access these activities is via the old right-click method on a Quick Toolbar category or item.

I do not plan on covering the included Dashboards as I think that you should be able to discover those capabilities on your own. What I am going to cover is the basics of creating your own Dashboards. So while you are reviewing the default Dashboards on your own, pay particular attention to the items contained within the Dashboard dialog form like multiple (tabbed) dashboards, the various Filtering capabilities. Many of these features may exceed my capabilities to cover them within this book, however, I will do my best to incorporate as many as possible.

New Dashboard...

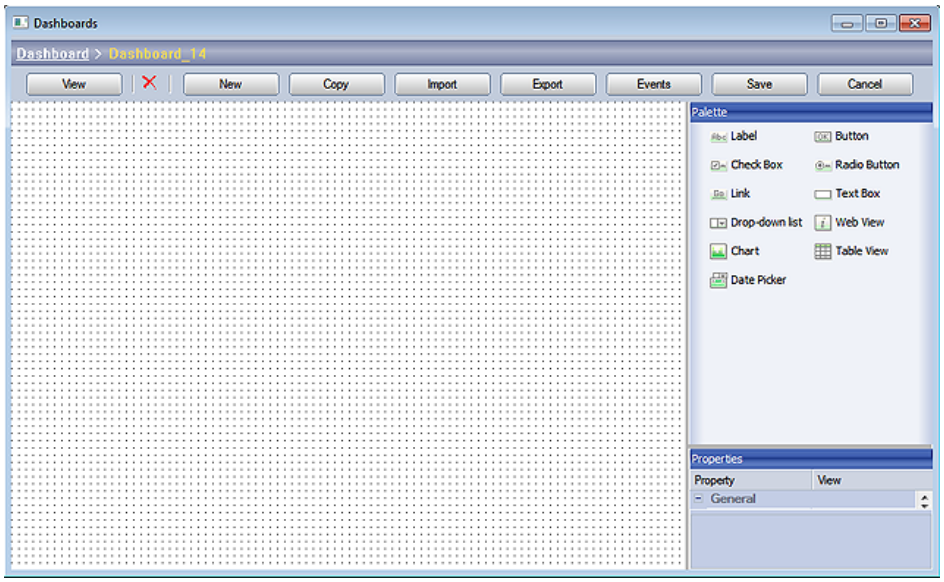


Figure 1-3

As the section title indicates, let's begin a **New Dashboard...**. Go ahead, and click on the **New Dashboard...** hotlink to bring forth the **Dashboards** designer dialog form as shown above in Figure 1-3 along with its' corresponding Quick Toolbar shown in Figure 1-4 on the next page. Again, pardon any distortion in Figure 1-3 as the compression was again quite severe. From here on I'll try to keep my screen shots to the segments of the dialog against which I am discussing.

Basically, as the screens that we will be discussing later in this book, this dialog form, the grid area in particular, represents a blank slate upon which we will be designing our dashboard(s). Referring to Figure 1-3, we have our general buttons above the grid area, and the **Palette** and **Properties** to the right of the grid area. The buttons permit you to **View**, **X** delete, create **New**, **Copy**, **Import**, **Export**, **Events**, and the standard **Save** and **Cancel** operations. On the other hand, your **Palette** contains the components that one may utilize in the creation of a dashboard, and we'll look at a couple of these a little later in this chapter. If you reviewed the default Dashboards, then you may have observed some of these components in action. They labels are pretty self descriptive so I won't repeat them all here.

Next we have the **Properties** container, which is an editible area that maintains some of the properties of your dashboard. As this area is compressed in Figure 1-3, I will list the default properties of **Category**, **Grid Spacing**, **Name**, **Timer** which are all in the **General** section, while the **Owner** property is part of the **Security** section. In turn, as you add components to your dashboard, this Properties container will change to reflect the properties of the component that was added or is highlighted.

Why don't we jump right in and change a couple of these properties. Let's change the **Category** to **Definitive Guide Examples**. We'll leave the **Grid Spacing** at its default setting, and we'll change the **Name** to **Definitive Guide Example 1**. That's it for this go around. If one were to click on the **View** button at this point, one would be able to see the new category of Definitive Guide Example along with its node of Definitive Guide Example 1. Voilà, you're a dashboard designer.

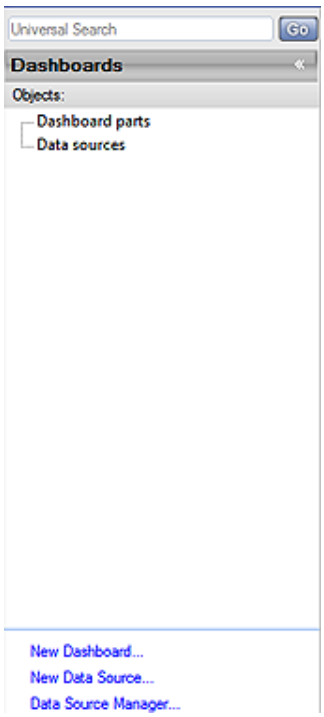


Figure 1-4

We need to add a few components to our dashboard, however, each component will eventually need to be linked to a data source. When you are looking at a dashboard, you are always looking at the component with its contained data source. Without this pairing you would have just a **Table View** say, without it having any data. Not a very good dashboard, is it? We control the various data sources by using the **Data Source Manager...** as shown in Figure 1-4, however, we will discuss that in a little more detail later in this chapter.

For now we just want to add a couple of components to give us a feel for how the components enhance your dashboards. Almost every dashboard will have at least two components, the **Chart** component allowing for the graphical representation of your bound data, and the **Table View** component allowing for the record presentation of the bound data.

Why not drag & drop the Chart component on the grid first placing it in the upper left hand corner of your grid leaving at least a 1 grid spacing to the top and left sides of the component. I then ask you to grab and drag the lower right hand corner handle such that your chart is approximately 6" wide and 3" in height.

Next, we'll want to modify a few of the properties for this chart. Did you notice that the **Properties** section changes when you dropped the Chart component on your gird? Before we forget, we would like to modify the **Anchors** which are contained in the **Layout** section of the **Properties**. And their definition, according to FrontRange is:

> **Anchors**
> Defines the edges of the container to which a certain control is bound.

For the purposes of this book, the container is our grid drawing surface while the control is what I refer to as the component. I suppose if I wanted be faithful to all of the designers among us, I would stick to the designers terminology, but I am trying to bring this book down to a more end user reading level.

Ah well, I wandered off on a tangent. By binding the edges of our component to the edges of our grid drawing space, we permit the component to anchor those edges such that if an end user changes the size of the dashboard, your component will change proportionally as well. Your design is maintained regardless of the Dashboard size set by the end user.

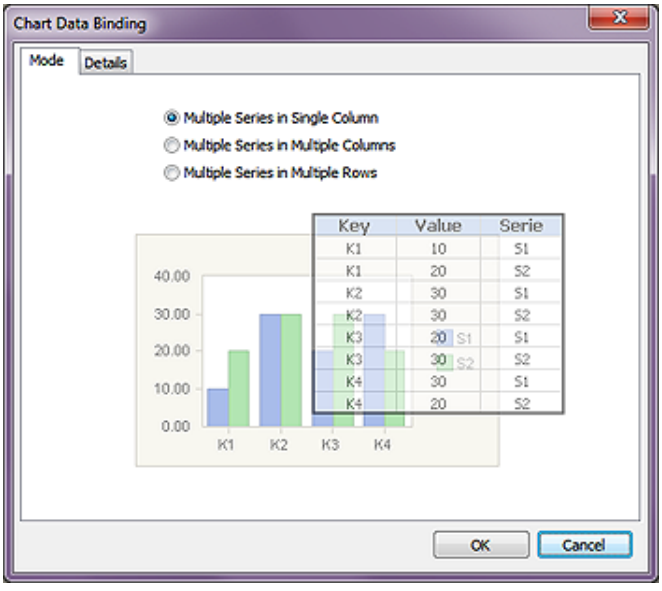


Figure 1-5a

I would like you to change the **Left**, and **Top** Anchors from **False** to **True**. For the time being we'll leave the **Right**, and **Bottom** Anchors at **False**. You may either do this on the single **Anchors** property (don't forget to Tab out when done) or via the individual properties below the **Anchors** property.

Without worrying about Data Management at the moment, we are now going to Bind our component to a data source. We can do this because the developers at FrontRange have predefined 20 categories of data sources, with each category containing

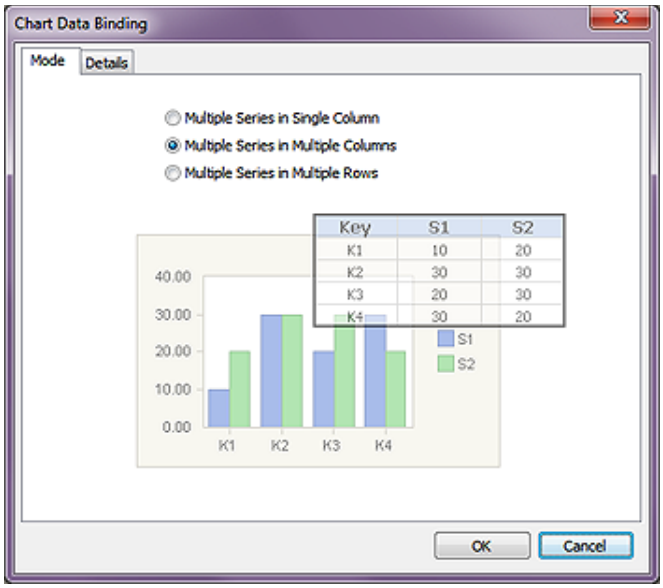


Figure 1-5b

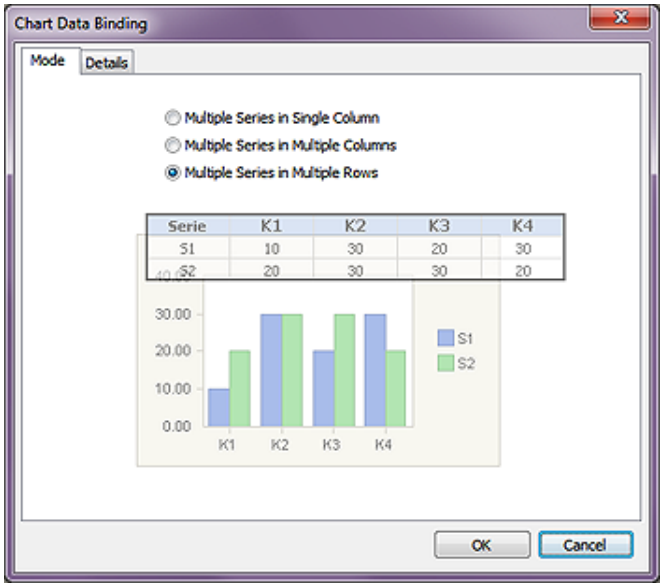


Figure 1-5c

multiple data sources. So let's walk through the minimal Binding process. In the **Properties** section, click on the ellipsis (**...**) to the right of the **Data Binding** property.

On the first tab page of the **Chart Data Binding** dialog form, we see that you have three radio button options from which to choose on the **Mode** tab. Notice that I have supplied an image of each screen as each option was selected, and pay particular attention to the displayed example table view associated with each radio button selection. This series of images is shown in Figures 1-5a on the previous page, and 1-5b & 1-5c on this page.

Although the GUI chart of the bound data never changes through the series of **Mode** selections, the table views of the selected data certainly do change. For this exercise I would like the default selection as shown in Figure 1-5a on the previous page.

Now, if you will, I would have you click on the **Details** tab of the **Chart Data Binding** dialog form which should result in the dialog form as shown in Figure 1-6 on the upcoming page. If you are not following in your GoldMine Premium, you may want to review the Figure 1-6 in this book at this time.

So in picking a category that is not too awfully complicated for this book, I have decided to utilize the **Data source category:** of **Closed sales**. Once I have made that decision I am forced to work within that constraint. For instance, the **Data source:** drop list was built based on our category decision, and contains some 8 possibilities in its default state of which I will select **Closed sales amount by user**.

This selection, in turn, populates the drop list for the remaining 5 selections. For the **Data are summarized by:** choice, I have selected **Activity_owner**. For the **Summarizing key legends (optional):** choice, I have again selected **Activity_owner**. Finally, in this section anyway, for the **Summarized values:** choice, I have selected **sum_amount**.

Now we have one more section to worry about or not (optional settings) on the **Details** tab of the **Chart Data Binding** dialog form, and that as stated on the dialog form is:

> **If the source contains multiple series of data, specify the following options.**

For both the **Series distinguishing field (optional):** and the **Series legends (optional):** choices, based on my previous selections, I have to leave them at their default state of **(empty)**.

If you have been following along up to this point then I would ask you to now click on the **OK** button, and then the **View** button in turn. Hopefully, you have been using GoldMine Premium as it was designed, and you users have Forecast and Closed their Sales. If so, your chart should look similar

to my chart as displayed in Figure 1-7.

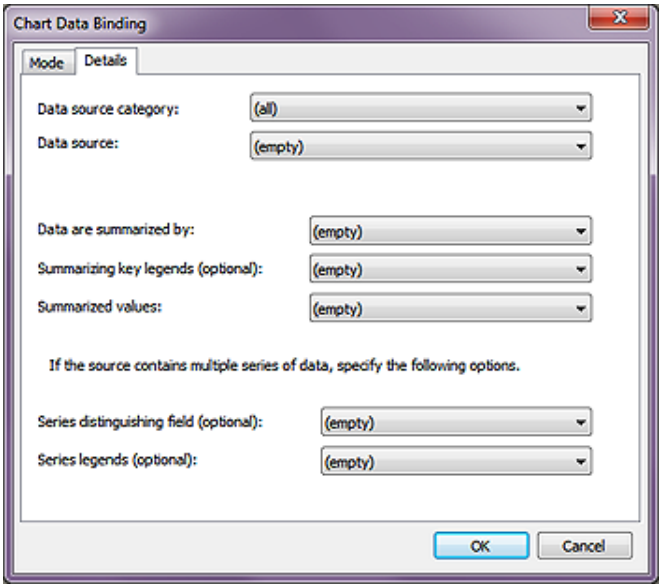


Figure 1-6

Looking at Figure 1-7, one might instantly notice that **BOB**, **LYNN**, and **JAMES** haven't had any sales. Visually, precisely and quickly, one can determine who the slackers are, and who the doers are in your organization.

What I immediately notice is that I require some sort of filtering capability because, in fact, Bob, Lynn and James are not even involved in sales within our organization. They are or were part of the Computerese Inc signature GoldMine Premium Support Consultant Team.

How about we add yet another component? Why don't you drag the **Table View** component below our **Chart** component leaving 1 grid spacing between the two while aligning the left edges. While we're at it, why don't we align right hand edges as well, and make the height approximately 4".

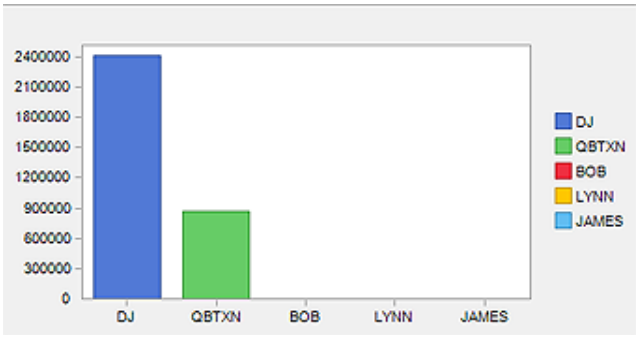


Figure 1-7

Here I have set the **Layout Anchor Properties** as **Left | True**, **Top | True**, **Right | False**, and finally **Bottom | False**. For the heck of it I also set the **Properties** | **General** | **Name** to **Closed Sales Table**. You should feel free to ad lib here if you wish.

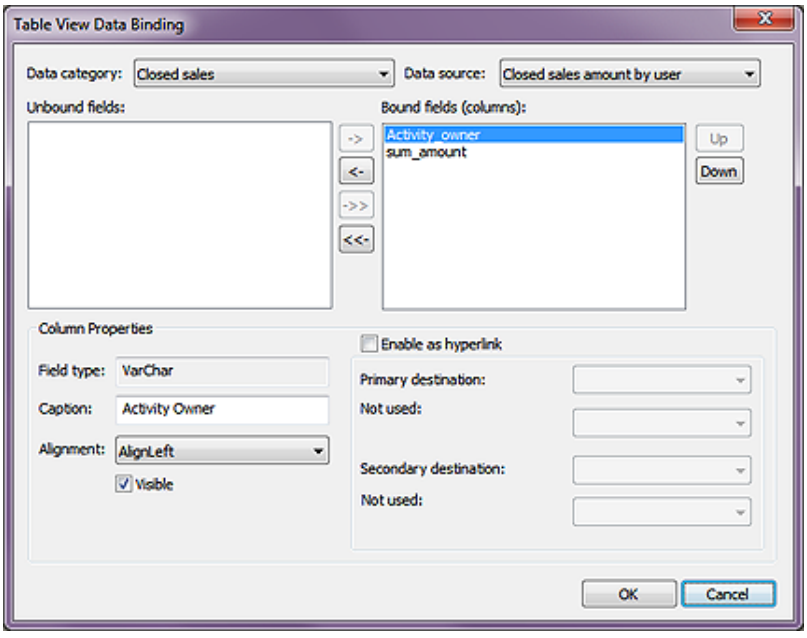


Figure 1-8

I think that we should begin to discuss the data binding for this component. We begin the binding process exactly as we did for the last component, however, this time the binding dialog form is dif-

Note

*Drag & Drop does not function between lists in the **Table View Data Binding** dialog form. To move all of the fields from one list to the other list would require that one click on either the **->>** or the **<<-** button whichever is appropriate for the direction you wish the fields to travel in.*

ferent. Notice, Figure 1-8 on the previous page, that the **Table View Data Binding** dialog form for this component is a single dialog form. We are going to bind this control to the same data source as we previously used because we want to display the data from which the chart was created for comparison purposes. Our **Data category:** will be **Closed sales**, while our **Data source:** will be **Closed sales amount by user** just as before. Immediately, **Activity_owner** and **sum_amount** will appear in the **Unbound fields:** list of fields. We want to move both of those fields over into the **Bound fields (columns):** list.

Clean up time. Next, we'll highlight the field **Activity_owner**, and change the **Caption:** for that column to a more dashboard display readable format of **UserID**, while we leave the column **Alignment:** value at its default for **VarChar Field type:** of **AlignLeft**. Additionally, we also want to leave the default value of ☑ **Visible**. We are not going to check the option to ☐ **Enable as hyperlink** as neither of these two fields links to anything that identifies the Contact record. Now let's highlight the **sum_amount** field, and change its **Caption:** to simply **Amount**.

Note

*Because of the nature of the data contained within this dashboard and depending on your organizations policies, you may want to consider changing this dashboards **Properties | Security | Owner** to that of a UserID or a User Group that you have established within your GoldMine Premium.*

UserID	Amount
DJ	2408653.49999999
QBTXN	873120.52
BOB	4871.5
LYNN	750
JAMES	375

Click on the **OK** button on the **Table View Data Binding** dialog form, and then again on the **View** button. Figure 1-9 shows you the results of the **Table View** once my data has been bound.

Figure 1-9

Did you notice anything unusual in Figure 1-9? Now I can clearly see that Bob, Lynn and James have all had sales. Graphically, because of the scale of DJs graph, their sales were not available in a visual manner where they are in the table grid. I bring this to your attention to show you a solid reason for utilizing a Chart and a Table View in your dashboards to show you the same data displayed in two unique formats. The more information that you can supply to your end users, the more power that they will have to wield.

Before I move us on to the next step I want to save a backup of this Dashboard. The way that we do that is by clicking on the **Export** button, and exporting this dashboard to a safe location. We'll restore from there later, so keep it handy.

Let's look at display a second screen within this Dashboard. The first approach that we will look at is the ability to subdivide the screen either horizontally or vertically into two independent screens. For this exercise I will work with the Vertical Splitter. With the Dashboard in the **Design** mode, right-click over the grid area, and select **Add vertical splitter** from the Local Menu. Now drag it to within one grid spacing from your existing components right edges. Technically, you now have two frames (screens) each independent of the other.

Want me to prove that? Drag the **Web View** component such that its left edge is one grid spacing to the right of the Vertical Splitter, and one grid spacing from the Dashboard on the three other sides. Under the **Properties** for this component, we will not do any **Data Binding**. I have changed the **Name** property on mine to **CNN World News**, and I have changed the **URL** property to **http://www.CNN.com/World/**. Lastly, I have set all of the **Anchors** to **True**.

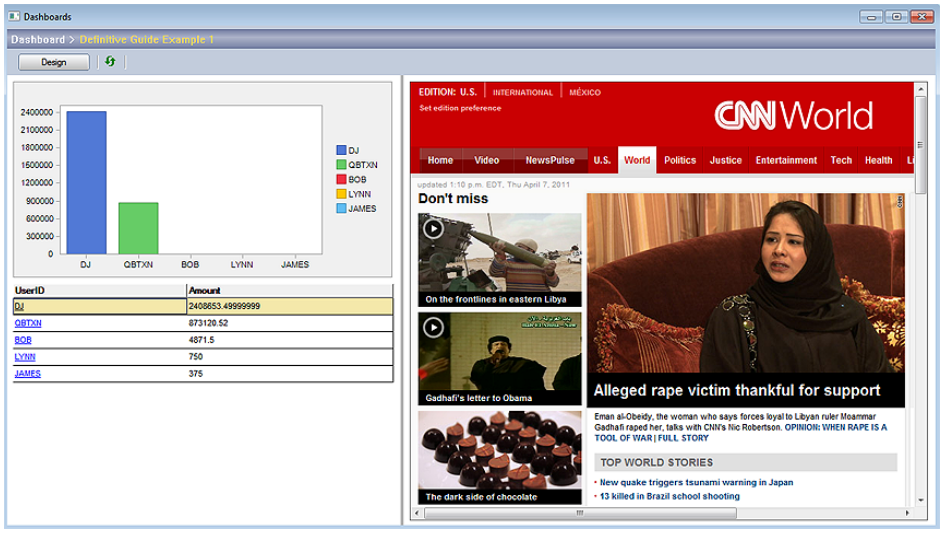


Figure 1-10

In Figure 1-10, you can see the consequences of my actions. And the acid test for independent frames (screens), drag the Vertical Splitter to the left and watch what happens. If everything worked

as expected the browser frame should have resized to the new width while the Graph/Table View frame should have gotten partially or totally blocked from view depending on how far you moved the Vertical Splitter to the left.

Let's look at another way to approach this. First we want to remove the splitter. I move my cursor over the splitter until it changes to parallel vertical lines with arrows pointing left and right. Next I right-click, and then I select the only option of the Local Menu the **Delete** option. When you are finished with that I want you to right-click over the **Web View** component, and choose **Delete control** from that Local Menu.

We've made it back to our base screen now so let's right-click over the grid area, and choose **Change to tab frame** from that Local Menu. I named my tab **Closed Sales by UserID**. I would then ask you to right-click just to the right of this tab, and select **Add tab** from the Local Menu. I named this tab **World News**. While this tab is active, add your **Web View** component just as we did previously. Click on the **View** button, and play around with your work. Again, we have two independent screens, but this time they each have their own tab, hence, they are visually separate.

Note

*To activate a Tab Frame, simply click on the tab, whether you are in the **Design** or **View** mode.*

Let's click on the **Design** button again, and this time I want you to click on the **Import** button. Because you did not delete the Dashboard that you were working on, you will receive the message shown here in Figure 1-11. Since this is only a practice run I would ask you to accept the default setting Replace existing dashboard, although, you could as easily have just created a new Dashboard from this import.

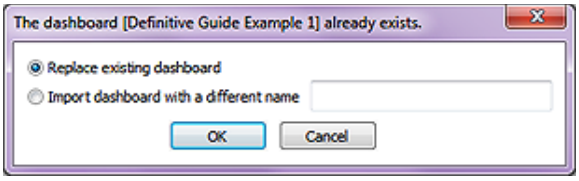


Figure 1-11

Let's try another component. Slide your Closed Sales Chart and Closed Sales Table down about 3/4" on the screen, and then Drag & Drop a **Label** component into the exposed grid area positioning it neatly. Some of the **Properties** that I change on mine are **Font | Tahoma, 18, Bold**, **Text Color | DarkBlue**, **Name | Closed Sales Title**, and **Text | Closed Sales by UserID**. Now, you'll need to switch between **Design** and **View** modes and stretch and heighten your **Label** component until your dashboard title looks correct and you are pleased with your design.

Tip

During the design stage, you want to remember to export your design after you have completed a stage with which you are satisified.

Backup Backup Backup

Again, just to be safe, I am going to Export this Dashboard, and you would be wise to do the same if you have been following along.

Component Binding

As of now I have 3 independent components on my Dashboard. Would it be great if the **Chart** component and the Table View component were bound such that when one clicks on a specific graph in the Chart component that the Table View reflects the data against which the Chart graph was built? I like to call this **Component Binding**, and within GoldMine Premium Dashboards, this is accomplished with the use of **Events**.

Note

A special thanks to Jamal Ahmad of FrontRange Solutions and Chris Wettre of W-Systems for their assistance in this area of my Dashboard education.

So let's begin by selecting and setting the focus to the Chart component. We can now click on the **Events** button to bring up the **Events and actions** dialog form shown here in Figure 1-12 although your **List of actions:** should be empty. Mine, on the other hand, is not empty as I have already added the **Drill down** action in preparation for this chapter in the book.

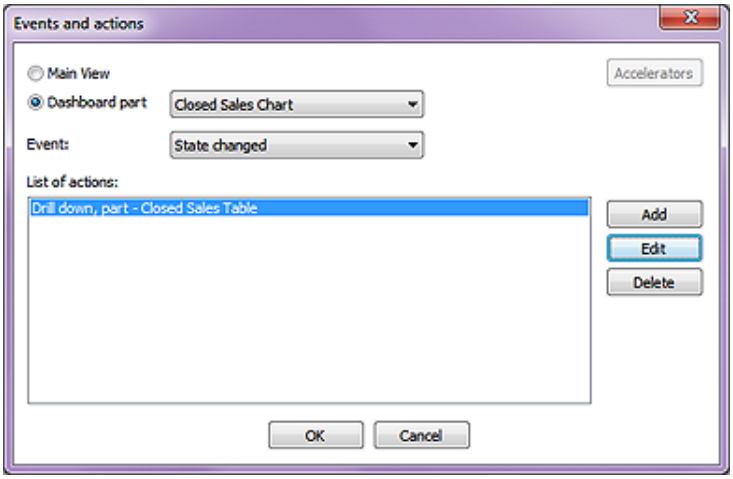


Figure 1-12

In fact, this is the very action that we need to add so I would have you click on the **Add** button while I will click on the **Edit** button. Figure 1-13 on the next page shows the **Action's properties, part - Closed Sales Chart** as I have already completed it in my pretesting for this book, however, I will go over each section as if I were creating it new.

First, notice that the dialog form **Title** has included the Dashboard component (part) **Name** property as developer information.

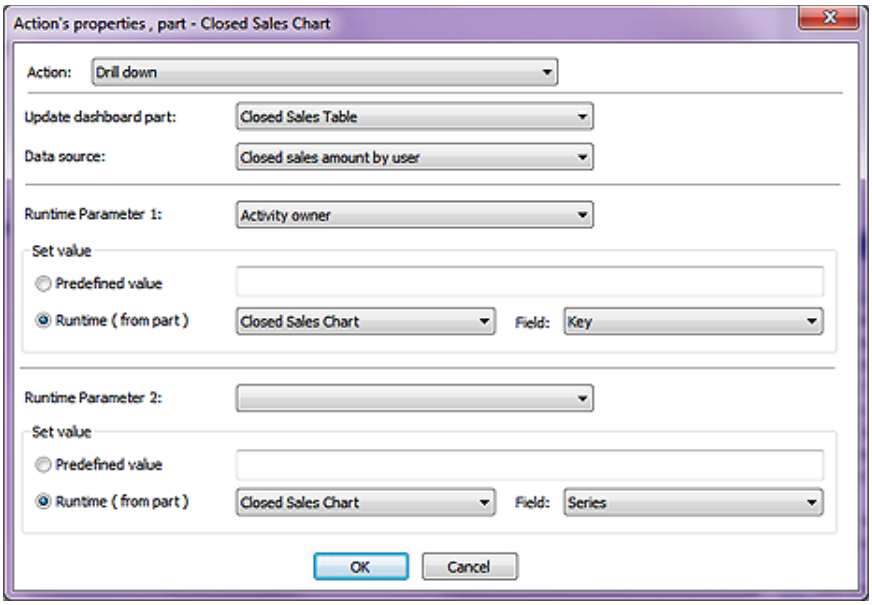


Figure 1-13

The **Action:** drop list is defaulted to **Drill down**, and as that is the action which I desire, I am willing to just accept it as is. One is next required to select the dashboard component which they would like to have updated as a result of this action, and this is done via the drop list for the **Update dashboard part:** field. You'll notice that only the components that you have already added to your dashboard will be listed in this drop list, and they will be listed utilizing the value as entered in their **Name** property. In this section of the dialog form you can also see the **Data source:** drop list, however, that is preselected for you this time as we had only selected a single data source for our **Closed Sales Table** component. I, therefore, graciously accept the **Data source:** of **Closed sales amount by user** for this entry.

Note

*The **Runtime Parameters** are predefined as part of the **Data Source**, and only those runtime parameters that were predefined for the selected data source will be made available to you via the **Runtime Parameter 1:** and **Runtime Parameter 2:** drop lists.*

For the **Runtime Parameter 1:** I have selected **Activity owner** from the available runtime parameters listed. Based on my selections so far the ◉ **Runtime (from part)** of **Closed Sales Chart**, and its' **Field:** of **Key** have been pre selected for me. Again, that is how I would have chosen my settings for this runtime parameter, however, I could have as easily chosen O **Predefined value** and added my own criterion.

Having done everything that is needed for this exercise, I would ask you to click on the **OK** button, and then on the **OK** button for the **Events and actions** dialog form. It might also be a good time to **Export** your Dashboard to another xml named file. I'm curious. Did you know that an xml is a structured document that contains name/value pairs representing your dashboard data in this particular case? Here's a the structure of the xml that I just saved:

```
<?xml version="1.0" encoding="UTF-16" standalone="no" ?>
<serialization xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:noNamespaceSchemaLocation="gmdashboard.xsd">

 <property id="View">
  <property id="Recid">FEWA1I5$H7B7R#Y</property>
  <property id="Name">Definitive Guide Example 1</property>
  <property id="Category">Definitive Guide Examples</property>
  <property id="Timer">0</property>
  <property id="Grid size">8</property>
  <property id="Owner"></property>
  <property id="Frames">
   <value>
    <object id="FramePanel">
     <property id="ID">frame</property>
     <property id="Parent_id"></property>
     <property id="Type">1</property>
     <property id="BaseArea">0 0 1168 865</property>
     <property id="MinSize">0 0</property>
     <property id="Visible">1</property>
     <property id="Pos">0</property>
    </object>
   </value>
  </property>
  <property id="Controls">
   <value>
    <object id="ChartView">
```

```
      <property id="Type">7</property>
      <property id="Frame">frame</property>
      <property id="Rect">8 64 568 328</property>
      <property id="InitRect">8 64 568 328</property>
      <property id="PrintRect">0 0 100 100</property>
      <property id="LpRect">212 1693 15028 8678</property>
      <property id="ID">FEWA1SW)=KPLR#Y</property>
      <property id="Name">Closed Sales Chart</property>
      <property id="Anchors">5</property>
      <property id="DataBinding">
       <object id="ChartMixedBinding">
        <property id="Data">EASPMA0%+/;\^Z!</property>
        <property id="KeyKeyField">Activity_owner</property>
        <property id="KeyVisualField">Activity_owner</property>
        <property id="SerieKeyField"></property>
        <property id="SerieVisualField"></property>
        <property id="ValueField">sum_amount</property>
       </object>
      </property>
      <property id="View Mode">2</property>
      <property id="3D Mode">0</property>
      <property id="3D Depth">20</property>
      <property id="Stacked">0</property>
      <property id="Gridlines">0</property>
      <property id="Color Scheme">0</property>
      <property id="Back Color">Control</property>
      <property id="Inside Color">Window</property>
      <property id="Title"></property>
      <property id="Keys">0</property>
      <property id="Keys Decimals">0</property>
      <property id="Values">0</property>
      <property id="Values Decimals">0</property>
      <property id="Keys Axis Title"></property>
      <property id="Values Axis Title"></property>
      <property id="Retain Source Order">1</property>
      <property id="Colored Values">1</property>
     </object>
    </value>
    <value>
     <object id="Grid">
      <property id="Type">6</property>
      <property id="Frame">frame</property>
      <property id="Rect">8 336 568 696</property>
      <property id="InitRect">8 336 568 696</property>
      <property id="PrintRect">0 0 100 100</property>
      <property id="LpRect">212 8890 15028 18415</property>
      <property id="ID">FEWA80E$4L#NR#Y</property>
      <property id="Name">Closed Sales Table</property>
      <property id="Anchors">5</property>
      <property id="Back Color">Window</property>
      <property id="Text Color">WindowText</property>
      <property id="Line Color">Black</property>
      <property id="Font">
       <property id="Height">-11</property>
       <property id="Width">0</property>
       <property id="Escapement">0</property>
       <property id="Orientation">0</property>
       <property id="Weight">400</property>
       <property id="Italic">0</property>
       <property id="Underline">0</property>
       <property id="StrikeOut">0</property>
       <property id="CharSet">0</property>
       <property id="OutPrecision">0</property>
       <property id="ClipPrecision">0</property>
       <property id="Quality">5</property>
       <property id="PitchAndFamily">0</property>
       <property id="FaceName">Tahoma</property>
      </property>
...

</serialization>
```

Of course I had to shorten it quite a bit to show you just this piece of the structure so that you could get a feel for how the xml document looks in its structured layout.

Any way, why not try your Dashboard now? First click on the **View** button if you haven't already. In the Chart, hover your cursor over one of the graphs, and watch what happens in the Legend, and the Table View. You'll see the Legend change before your eyes, however, the Table View remains constant.

Changing Default Dashboard Properties

Next click on that same graph in the Chart, and this time the Legend won't change (unless you move the cursor a hair after you click the graph), however, the Table View now displays only those records against which that particular graph unit was constructed.

I think that I'm going to stop diddling with this Dashboard for the time being so that we can move ahead. I want to look at some of the changeable properties on the default Dashboards. True, there aren't many over which you have any control, but there are a few.

Expand the **Management** branch in the Quick Toolbar if it is not already expanded. Selected any default dashboard, let's say the **Activity Dashboard**, and right-click to bring up the Local Menu. Next I want you to select **Properties** from that menu. The **Specify Dashboard's name** dialog form, as shown in Figure 1-14, becomes available, but not only can we modify the **Name:**, we could also move it into a different **Category:** or even create a totally new category here. Lastly, we have the ability to assign an **Owner:** which could be a specific UserID or a User Group should you have any predefined.

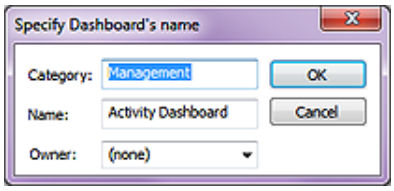


Figure 1-14

As is clearly shown, the default **Owner:** is **(none)** such that everyone has access to this Dashboard. This may be particularly useful for the **Forecast Dashboard** or the **Opportunity Dashboard** where you might want to permit only managers access to this information, remembering in the back of your mind that Master Rights users trump all settings.

Moving forward now, right-click on any bar in the **Activity Dashboard** to bring up the Local Menu which should contain:

Gallery ►
Color ►
Point Labels
Properties...

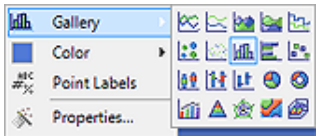


Figure 1-15

If one were to actually hover the cursor over the **Gallery ►** option, one should see the various graphs available for displaying the associated data. You can play around with this as you want to see the various types of graphs that are made available to you and how they appear in action.

I think that **Color ►** is fairly obvious as to what it represents, however, maybe **Point Labels** are not as obvious. If one were to select **Point Labels** while in the Local Menu for one of the bars or pie pieces you would see the labels appear over or on the various pieces of the charts as shown here in Figure 1-16.

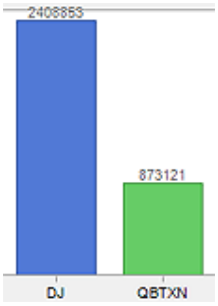


Figure 1-16

If one were to right-click off of the actual graph, but still on the Dashboard, one might see that they have the option of changing:

Toolbar
Data Editor
Point Labels
Gallery ►
Color ►
Edit Title
Point Labels
Font
Properties...

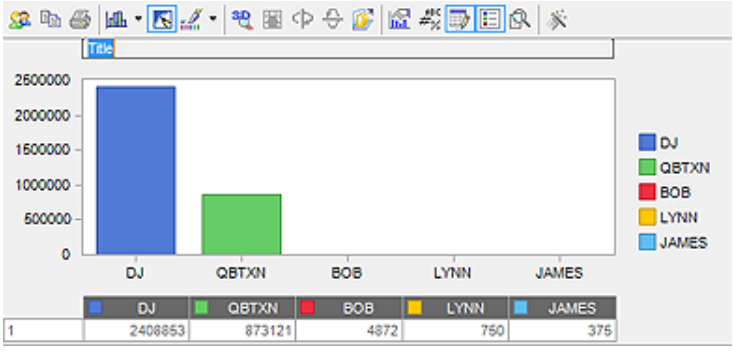


Figure 1-17

One will find that some of these settings are sticky, while some are only sticky to a point. Let's say that one selects **Data Editor**, **Edit Title** or **Toolbar**. Ones chart might look like the one shown above

Note

*Changing the graph through the **Gallery is** sticky. In other words, if you were to change a bar graph to a pie graph, leave that Dashboard for another, and return to the original Dashboard the pie graph style would have been maintained.*

Note

*Editing the Title of the default Dashboards **is not** sticky. Doing this will change the Title, however, only while you are utilizing that instance of the Dashboard.*

in Figure 1-17. And, one you may certainly change the title or any of the individual numbers that make up each graph unit. One could also **Print** or **Copy to Clipboard** this chart at this instance in time as manipulated by hand, however, click on any other dashboard and then on this dashboard again, and those edits that you had made will be gone completely. True the **Data Editor** and **Toolbar** will still be displayed there, but all of your modified entries will be gone.

Again, you'll want to play around (aka Trial & Error) with these features to see how they could be best utilized for your needs. In particular, a good area to play is with the **Properties...** dialog form as there are many display (beautification) items that can be modified in this area.

I don't think that I'm going to go into the functioning of the Grids here as they function as any grid within GoldMine. One could **Sort**, **Filter**, **Group**, etc on the grid to arrange the data in any of a number of presentations, and then as simply **Output to Excel** that very data among other avenues.

Data Source

Earlier I promised you that we would take a closer look at the **Data Source** feature of any **Dashboard**. It's about time we did just that. *What is a Data Source*, you might ask. Well to me, in Visual FoxPro terminology it is a Cursor which, in turn, is the result of executing a SQL Query. You have seen SQL Queries before within GoldMine maybe, if not, I will be discussing them in detail later in this book in Chapter 7 - **Gathering the Data**. For now let's look at:

Note

*Most of the Dashboard Options will not be available unless one of your dashboards is placed in the **Design** mode. You'll need to remember that throughout this chapter.*

Tools
SQL Query

We could look at any query at this point, however, for consistency, why don't we all look at this query:

```
select C1.AccountNo,
  C1.Company,
  C1.Contact,
  C1.Key1,
  CH.UserID,
  CH.OnDate,
  CH.Ref as [Reference]
from Contact1 C1,
  ContHist CH
where C1.AccountNo = CH.AccountNo
  and CH.RecID in
    (
    select max(RecID)
    from ContHist
    where sRecType = 'A'
    group by AccountNo
    )
order by CH.OnDate desc,
  C1.Key1
```

If you were to run this query, the resulting cursor would display the last appointment that was had with your respective contacts.

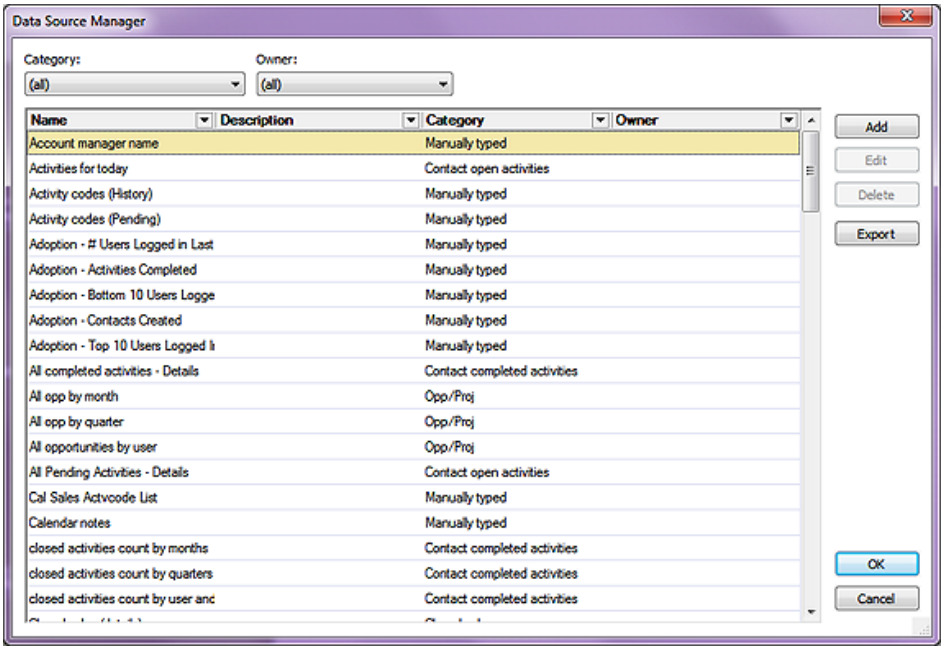


Figure 1-18

This is basically, what a Data Source is, and we have also learned that components in our dashboards are tied into a data source through the components properties.

While we are in the Design mode for our Dashboard, and in the Quick Toolbar to the left, let's click on the hotlink for **Data Source Manager...** to expose the **Data Source Manager** dialog form shown on the previous page in Figure 1-18.

Note

*Even a GoldMine User possessing Master Rights will not be able to **Edit** or **Delete** any of the GoldMine default Dashboards.*

From here it is easy to see that one may filter down the list by **Category:** or **Owner:** or both for that matter. Furthermore, we could **Add**, **Edit**, **Delete**, or even **Export** a data source.

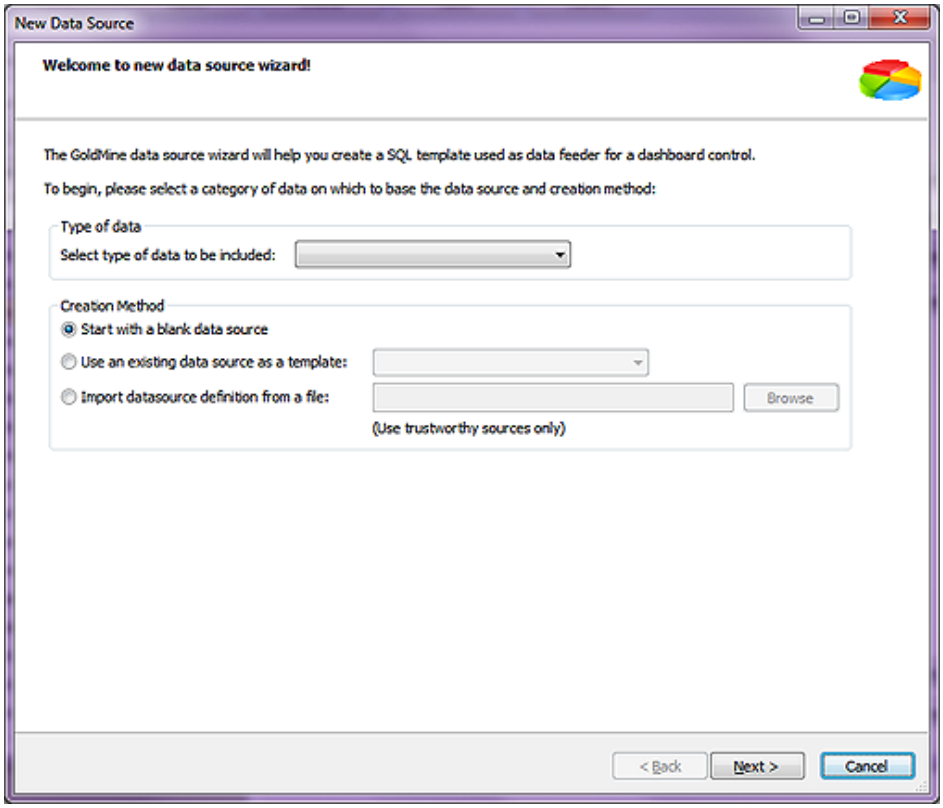


Figure 1-19

Note

I'm going to follow, in written form, the videos developed by Iain Wicks for this area so that you can always refresh yourself by using that video if you have it.

Let's begin now by building a Data Source of our own by clicking on the **Add** button to produce the **New Data Source** dialog form shown above in Figure 1-19. You should immediately notice two separate and distinct frames, the **Type of data** frame and the **Creation Method** frame.

If you focus on the first frame for the moment, you see that you have but one choice and that is the **Select type of data to be included:** which consists of a drop list of values to which you may not add to or edit any of the items within the list. Upon closer examination of this list, you might have noticed something familiar. Have you noticed that this list of items look suspicious like the Category list which you saw earlier? The observant among you will have. For this exercise let's just select the **Contact Info** category.

In the **Creation Method** frame you see that you have radio button options with the first being ◉ **Start with a blank data source** which just so happens to be the GoldMine default selection, and is the Creation Method that we will utilize in just a bit. The next option is to ○ **Use an existing data source as a template:** and is always a good place to start if you have an existing data source of the type that you need. While the final Creation Method, remember that **Export** button that I mentioned earlier, is to ○ **Import datasource definition from a file:** and one could then **Browse** for said import file. Of course, as stated, **(Use trustworthy sources only)**.

Let's trudge on down the road then, shall we, and click on the **Next >** button which will take you to the **General properties** dialog form of the Wizard as shown in Figure 1-20 at the top of the next page. You will notice that there are a few data fields and the **Output type** frame. The **Name:** field is simply for entering the name that you wish to remember this Data Source by later on when you select it from your list. You'd be wise to make it as descriptive as you can so that you have a chance of remembering the type of data which it pulls for you a year from the time when you create it and its function is clear in your mind. Ah, that goes without saying I guess, but I had to say it any way. You probably see me restate that a number of times throughout this book. Because we selected Contact

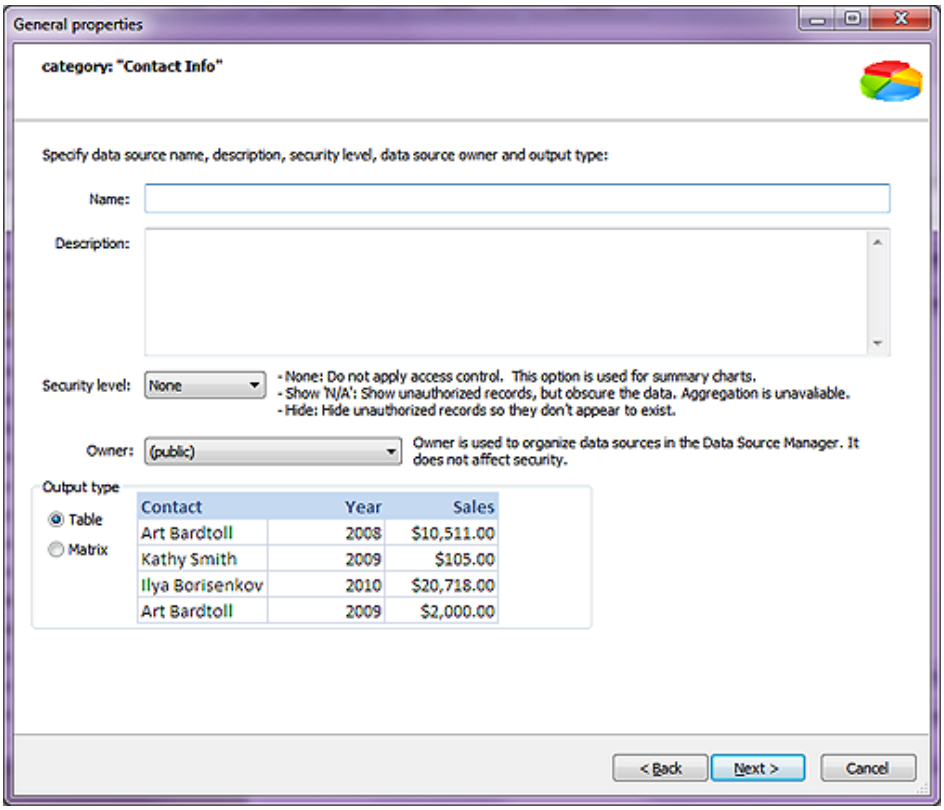


Figure 1-20

Info as our Category previously, why not just give this data source a **Name:** of **Contacts Information with Source Field**. I ask for this name as later on I plan to tie this data source in with another data source that will count against the Source field.

You can enter a **Description:** if you want to, and I highly recommend that you do that. Again, when you look at these a year from now, you are going to want to have as many ticklers to your brain cells as possible to help you in remembering exactly why you created this in the first place.

Next, you have the **Security level:** option, and FrontRange actually supplies you with a pretty good definition of this directly on the dialog form:

> **- None: Do not apply access control. This option is used for summary charts.**
> **-Show: 'N/A': Show unauthorized records, but obscure the data. Aggregation is unavailable.**
> **-Hide: Hide unauthorized records so they don't appear to exist.**

If you remember when I talked about Properties earlier, you could control who can view the dashboards by controlling that through the Properties of each dashboard. For this exercise, why not just leave this at its default setting of **None**.

Where next we come upon the **Owner:** field. Unlike the Property Owner, this Owner field does not affect security one bit. It is simply a means of organizing the Data Sources in the Data Source Manager. Odd that FrontRange would use such a security oriented label for an organizational task.

So now we have a radio button option in the **Output type** frame, and there are but two choices. A ◉ **Table** or a ○ **Matrix** (not to be confused with the cult movies). The **Table** type of output is nothing more than Columns identified in Row 1 which acts as the Header for the Columns, and Rows (Records) which contain the data delivered via the Data Source. The sample data for this type is displayed in Figure 1-20 above. Simple enough, and something with which a GoldMine user should be more than familiar.

On the other hand you might not recognize a **Matrix** as a Pivot Table or what some may call Cross Tabbing Table. As you may not be as familiar with this type of output I have used the GoldMine example here in Figure 1-21 to help you better visualize this output type. You'll notice that the Se-

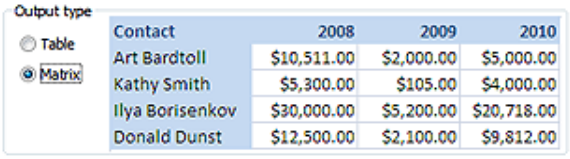


Figure 1-21

Tip

*Whenever you create a Data Source, remember to select your UserID as the **Owner:** of the Dashboard to aid in locating it later among the many Dashboards.*

ries is include in Row 1, the Header Row, while the Rows contain the Series Data as gathered by the Data Source.

You should now want to click on the **Next >** button to proceed to the **Select** dialog form of the Wizard as shown in Figure 1-22.

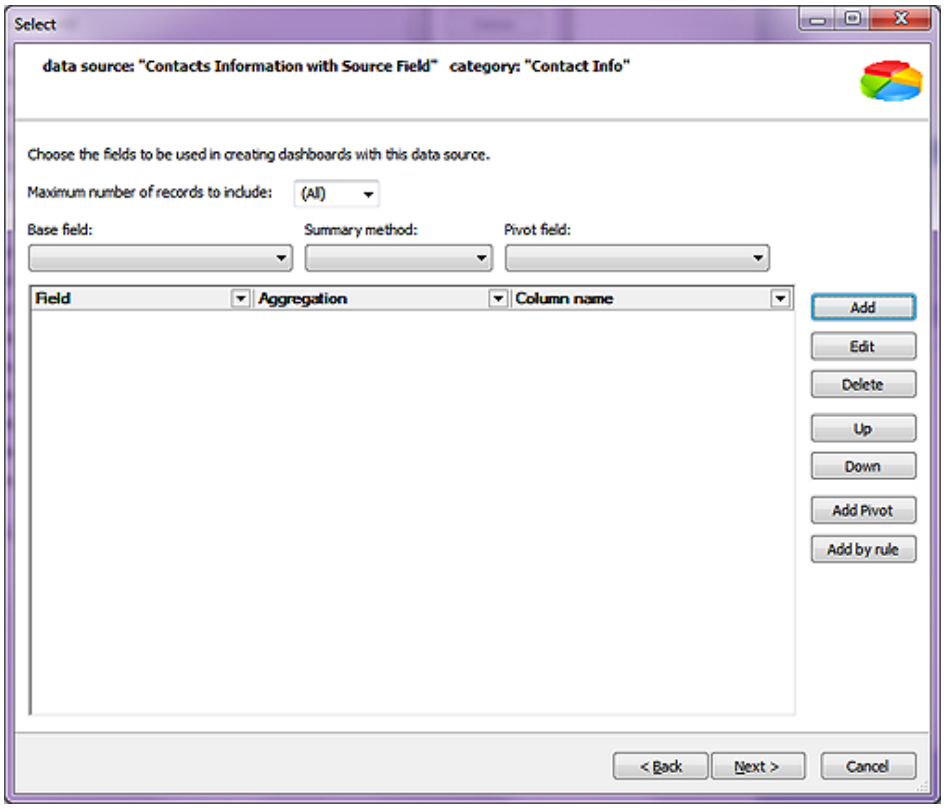


Figure 1-22

This is where you would choose the data that would be included as the **select** segment of your SQL Query. Some of the options are clearly displayed in Figure 1-22, but include the ability to **Add**, **Edit**, **Delete**, Move **Up** or **Down** in the selection order, to **Add Pivot**, or to **Add by rule**. By clicking on the **Add** button, we are presented with the **Add/Edit field** dialog form shown in Figure 1-23.

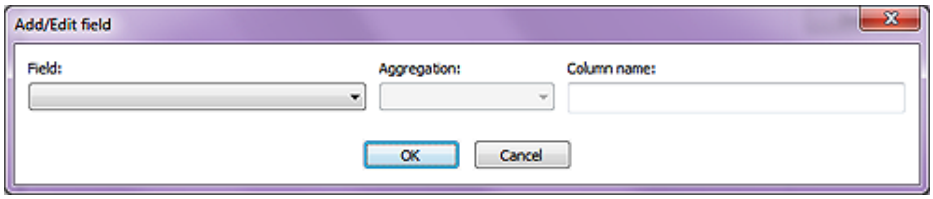


Figure 1-23

It is here that one could, obviously, Add a new selection or Edit an existing selection statement. The **Field:** drop list contains common name macros towards the top of the list like **Company**, and alias.field names like **CONTACT1(c1).COMPANY** further on down the list. As we selected a Category of Contact Info earlier, the **Field:** drop list was built with Contact1 and Contact2 fields or macros only. Here is the list of fields and their column names that I am proposing that you add for this example:

Field:	Column Name:
CONTACT1(c1).COMPANY	Company
Primary contact	Primary Contact
CONTACT1(c1).SOURCE	Source
CONTACT1(c1).KEY1	Contact Type
City	City
Phone	Phone

You'll notice that I have select to use some of the macros and some of the alias.fieldname fields so that you could get a feel for their workings. We can make the Column Name: anything at all that we desire or that would be meaningful to the dashboard viewer. Once you have added those fields, you'll want to click on the **Next >** button if you are following along in this example.

Note

*For this Data Source, I ask you to forget about the **Aggregation** value, although, I promise you that we will utilize that later.*

*All that you need to know at this moment is that the **Aggregation** value selection permits:*

AVG
COUNT
MAX
MIN
SUM

Hopefully, these are self-explanatory terms.

And doing so should bring you to the **Predefined Search Criteria** dialog page of the Wizard which you may better recognize as a SQL Query filtering condition or the **where** segment of a SQL Query, Figure 1-24.

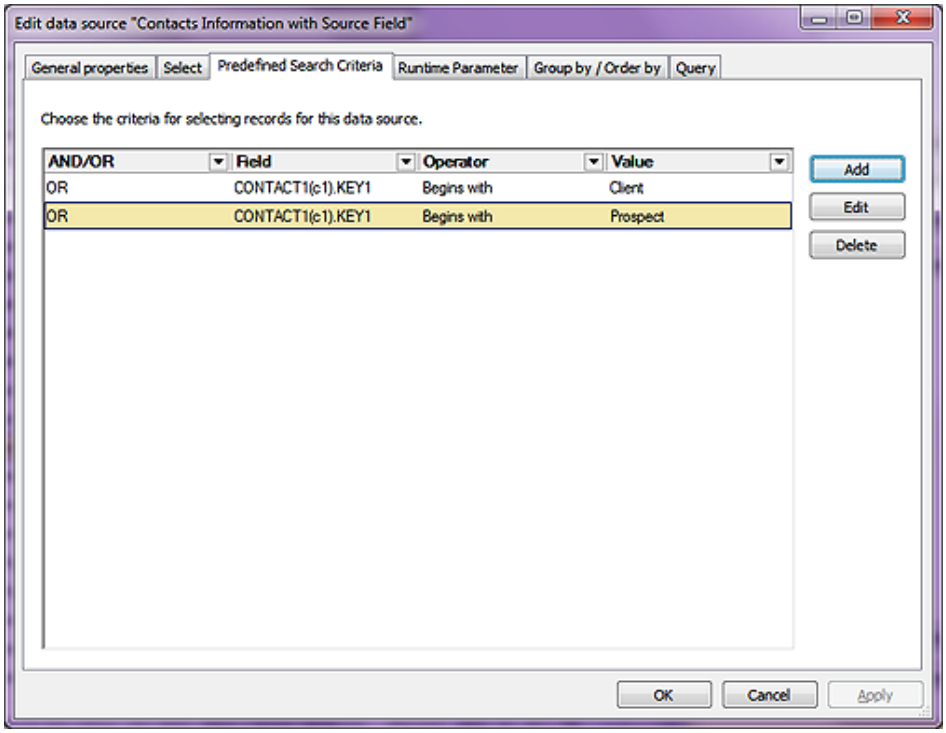


Figure 1-24

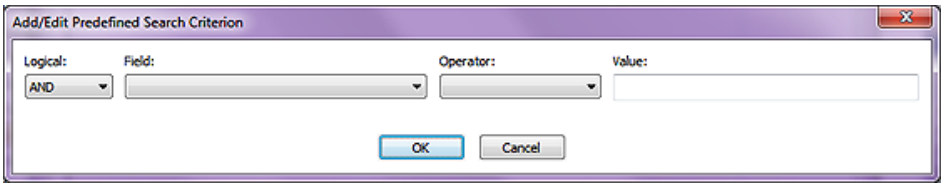


Figure 1-25

Since I have Vendors and GM Dealers in my **Contact1.Key1** field among others, I want to target this search to just my Clients and Prospects. Unfortunately, the **Value:** field, as shown in Figure 1-25, does not contain the F2 Lookup List as we're used to when creating filters such that you must manually type in the **Value:** so be extremely careful to check your spelling.

That is all that I need to add for the **Predefined Search Criteria** dialog page of the Wizard, and if I weren't now in the **Edit** mode, I would click on the **Next >** button as you should. However, because I have switched to the **Edit** mode, I will click on the **Runtime Parameter** tab so that I will be on the same dialog form as those of you following this exercise.

Note

*Please remember, from here on out through the rest of this section of this chapter, that my screenshots are taken from the **Edit** mode and will be slightly different from your Wizard displayed dialog form.*

I will be discussing them, however, as if I were still in the Wizard.

These figures are so large that it's hard to fit many on their proper page in a book of this size, so the **Runtime Parameter** dialog form is being shown in Figure 1-26 on the next page. Looking at what FrontRange has to say about this page of the Wizard, we can see that *"Runtime Parameter enable end-users to select the records to display in a dashboard. You can choose them on this page."*

Other than the crappy English, do you buy that statement? I don't, at least not as stated. We have already selected, via previous dialog form pages in this Wizard, everything that we actually want such that this is more of a staging area that the developers use to provision their data source selection code. True, one can add more runtime parameters or even modify or remove existing runtime parameters, yet for the most, you will leave this dialog form untouched.

These are normal runtime parameters that the GoldMine Wizard will add to each and every dashboard created via the Wizard. On the other hand, and for this exercise, I would like you to click on the **Add** button, see Figure 1-27 on the next page, as we are going to add a runtime parameter. Do you remember earlier that we named this dashboard: **Contacts Information with Source Field**? Well we are now going to add a runtime parameter for Source.

You'll notice in Figure 1-27, that I have already pre filled in my desired information, but I would like to explain each selection.

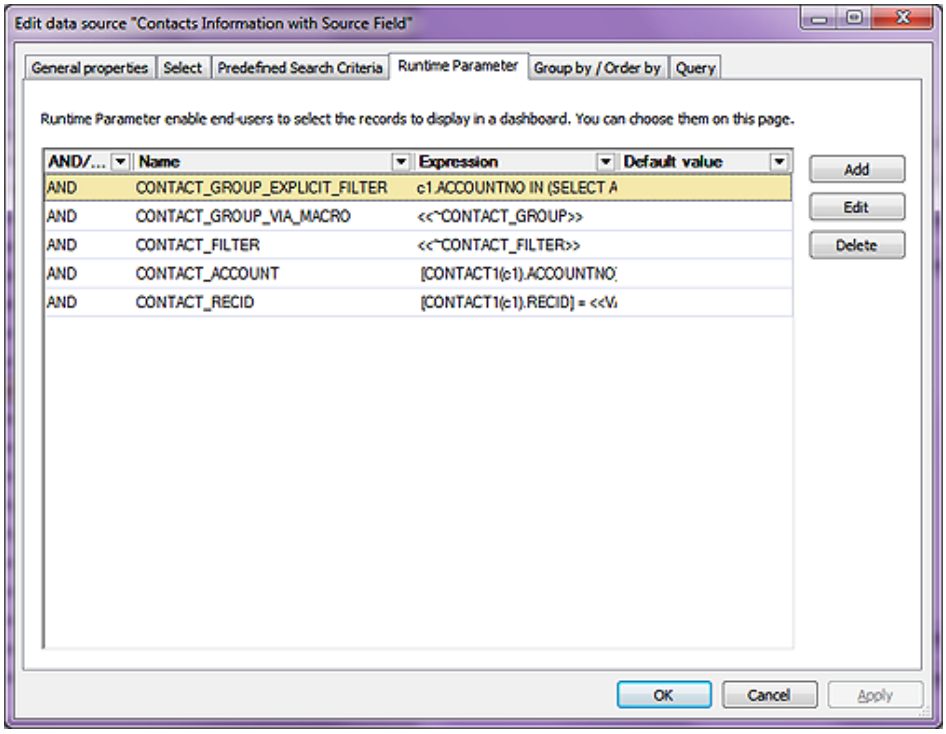


Figure 1-26

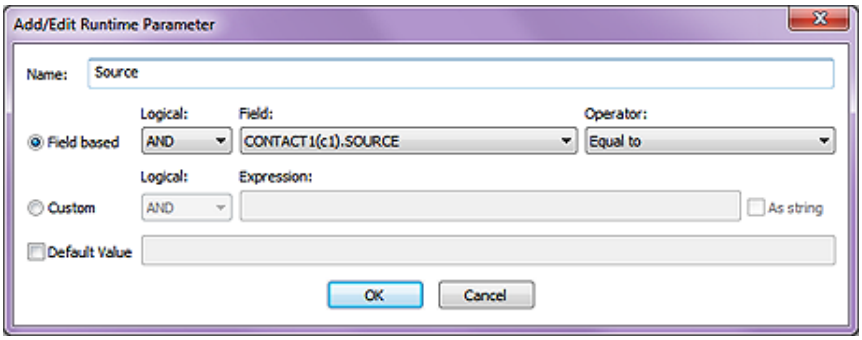


Figure 1-27

In the **Name:** field I simply ask you to enter **Source** which is a clear enough name for our runtime parameter in that I'll remember what that is next year if I rewrite this book.

I want you to accept the default ◉ **Field based** option at this point, however, you could have just as easily selected to create a ○ **Custom** parameter or set a ☐ **Default Value**. You notice, Figure 1-26, that all of the default runtime parameters were established with **Logical: AND** instead of **OR** so I see no reason for not following suit. For the **Field:** I would like you to select **CONTACT1(c1).SOURCE**, and we accept the default **Operator:** of **Equal to**. A simple click on the **OK** button, and your runtime parameter will be added to the existing list of runtime parameters.

Now the question might be: "Why did we do that?", and the answer to which is that, later on, I may add a drill down on our Chart based on the Source, and we would want the Table View to reflect only that data for the selected Source. You'll see as we progress, but suffice it to say that we will need it for this example exercise.

Click on the **Next >** button, to bring up the **Group by / Order by** dialog form of the Wizard, Figure 1-28 on the next page. I think that the GoldMine explanation is worth repeating on this one: *"Specify how data will be grouped and sorted. You can choose among fields included to SELECT."*.

That's as good an explanation as any, however, I will ask you, for this example, to not use the **Group by:** section of this dialog form as grouping is unnecessary for this example. On the other hand, I would like you to use the **Sort by:** option. On the **Sort by:** drop list, I would have you select **Company**. **Do not** leave this screen as yet. If you do not click upon the **Add** button at this point, then the **Company will not** be added to the **Sort by:** listing, hence, the query. Click that **Add** button now please, and the click on the **Next >** button to proceed to the final dialog form of this Wizard which is the **Query** screen as shown in Figure 1-29 on the next page.

Note

An example of what a grouping and sorting statement might look like in a typical SQL Query might be:

```
select c1.Source,
   count(*) as [Count]
from Contact1 c1
group by c1.Source
order by Count desc
```

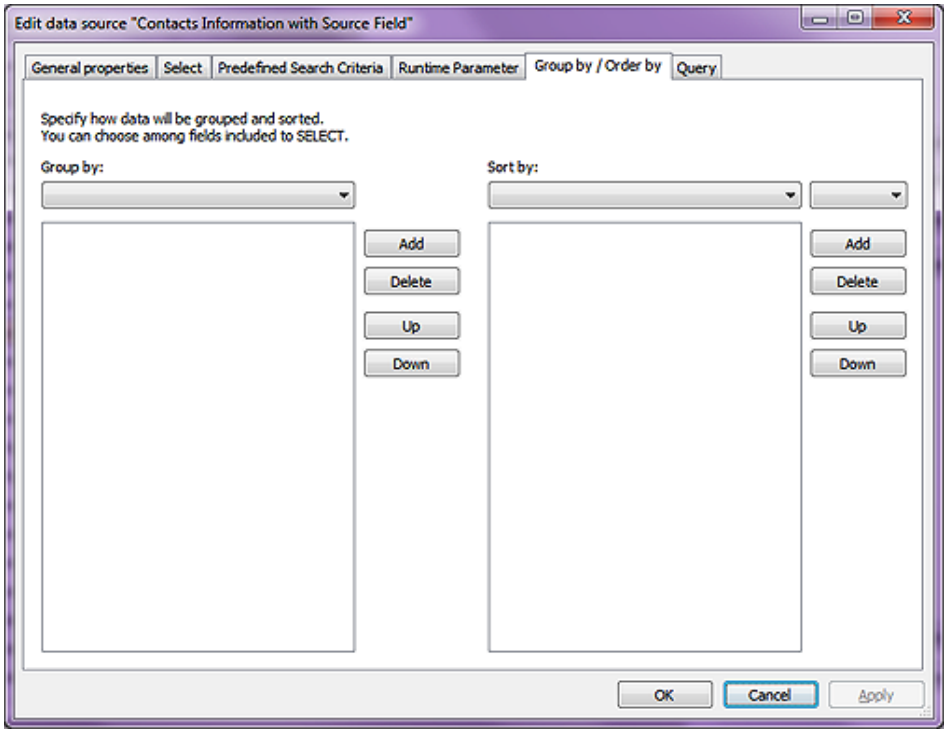


Figure 1-28

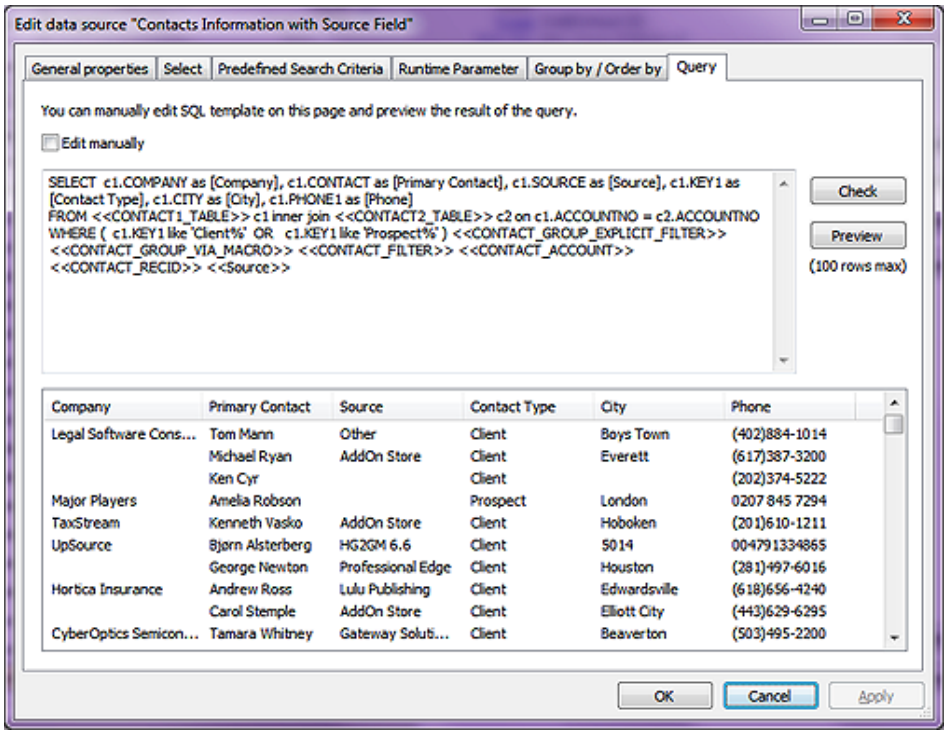


Figure 1-29

You will notice that your dialog form, at this point anyway, will not look quite like the one shown here in Figure 1-29. You see, I did two things to get it to this point. First I clicked on the **Check** button to check the integrity of our query. As we utilized the Wizard to build this query, it should have checked out just fine, and it did. So why is the **Check** button even here? Well, technically we could edit this query manually causing that integrity to become unstable. Notice the ☐ **Edit manually** checkbox just above the query.

As for the next step, I clicked on the **Preview** button to make sure that the query was pulling the correct data, and it appears to be doing that just fine. At this point, you could click on the **Finish** button, or I on the **OK** button, and I think that we should do just that.

Creating a New Dashboard

That's it. You've created your first **Data Source**, and it is ready for insertion into a Dashboard.

What say we put our new data source to some good use by creating a **New Dashboard...**. Go ahead, click on the **New Dashboard...** link in the Quick Toolbar, and then let's change the property of **Category** to **Definitive Guide Examples** as you had done previously. Why not change the **Name** property to **Definitive Guide Example 2** while you're at it.

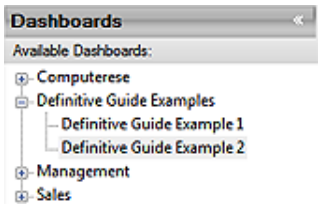


Figure 1-30

You won't need to do this as frequently later, but because you are still new at Dashboards, let's have you click on that **View** button to make certain that you're good so far. Your Quick Toolbar tree should look similar to mine as shown in Figure 1-30, and if it does, you should click on the **Design** button in preparation for moving forward on this example.

Let's go ahead an Drag & Drop a **Table View** on there now, and stretch it out to give it some size. Let's change the **Properties** just as we did in the **Changing Default Dashboard Properties** section of this chapter only this time I would have you make the **Table View Name** property **Contacts by Source**. Additionally, I would like you to change the **Data Binding** property to your new **Data Source**. Make certain that you throw in all of the columns which is most easily done by highlighting the first column in your **Unbound fields** list, and then clicking on the **->>** button to move them all at one time into the **Bound fields (columns):** list to the right. And after you have done all of this, and clicked on the **OK** button, please click on the **View** button again. If everything was correct, you should see your data in the grid just as it was seen earlier when you created your **Data Source**.

Let's click on that **Design** button once more, and make some additions. I want you to create a new **Data Source** on your own this time. I'll just give you the settings that I would be looking for you to include in this data source. The **Type of data** will again be **Contact Info**. This time you'll give it a **Name:** of **Count of Contacts by Source**, and set yourself as the **Owner:**. As well, you'll accept the default **Output type** of **Table**. For this **Data Source** you'll only require two columns, and the first of which should be **CONTACT1(c1).SOURCE**. Here's a new one for you, however, you are going to add a macro as the second column which you have not encountered as yet. Please add the **Items Count** macro as your second column. As you'll find out, this is basically an automated **Group By** statement.

Moving ahead now, please add the same two **Predefined Search Criteria** as you added when making your first **Data Source**. You won't need to add any **Runtime Parameters** as this is going to be the **Data Source** for a **Chart** on the dashboard that you are designing. While you are on the **Group by / Order by** dialog form of the Wizard, I want you to notice that the previously selected macro is now appearing in the **Group by:** listing automatically. When you get to the **Query** dialog form page of the Wizard, your query should look similar to this, only it will be unstructured:

```
SELECT  c1.SOURCE as [SOURCE],
  count(*) as [Items_Count]
FROM <<CONTACT1_TABLE>> c1
  inner join <<CONTACT2_TABLE>> c2 on c1.ACCOUNTNO = c2.ACCOUNTNO
WHERE (  c1.KEY1 like 'Client%'  OR   c1.KEY1 like 'Prospect%' )
  <<CONTACT_GROUP_EXPLICIT_FILTER>>
  <<CONTACT_GROUP_VIA_MACRO>>
  <<CONTACT_FILTER>>
  <<CONTACT_ACCOUNT>>
  <<CONTACT_RECID>>
GROUP BY c1.SOURCE
```

Does yours look something like this? I hope so. Click on the **Check**, and then the **Preview** button to see a cursor result set as extracted based upon your query. Click on the **Finish** button to wrap things up properly.

Drag & Drop a **Chart** onto your designer screen just above your **Contacts by Source Table View** much as we did before when creating a new dashboard. You should change the **Name** property to **Contacts by Source** as **Chart Name** won't be confused with your **Table View Name** as they are two separate and distinct components. Also, don't forget to change the **Data Binding** property. This time you'll want to set it up as I have done in Figure 1-31 on the next page.

Now go ahead and click on the **View** button to view the results of your working so far. You should have a properly functioning **Chart**, and a properly functioning **Table View** albeit functioning independently of each other. If you're with me so far, that's exactly where I want you to be.

At this point I'm going to have you make a modification to your **Contact Information with Source Field Data Source**. I would like you to **Sort by:** the **Company** field just in case you use Iains Video to watch what I am discussing in this book.

Note

*Have you noticed all of the extraneous statements in your query? For instance, you have no need for information out of the **Contact2** table, yet the Wizard has included an **inner join** statement joining the **Contact1** and **Contact2** tables on their respective **AccountNo** fields.*

Note

I had to do just this before I began writing this chapter. I had to contact FrontRange for instructions on exactly how to accomplish this. Alas, there was nothing written on this, and very few knew how to even design a dashboard.

Remember now that we have two independent components functioning in our dashboard, each using its own independent data source. Wouldn't it be nice if one had the ability to click on a graph unit on the chart and have the data that comprises that graph unit display below in our table view?

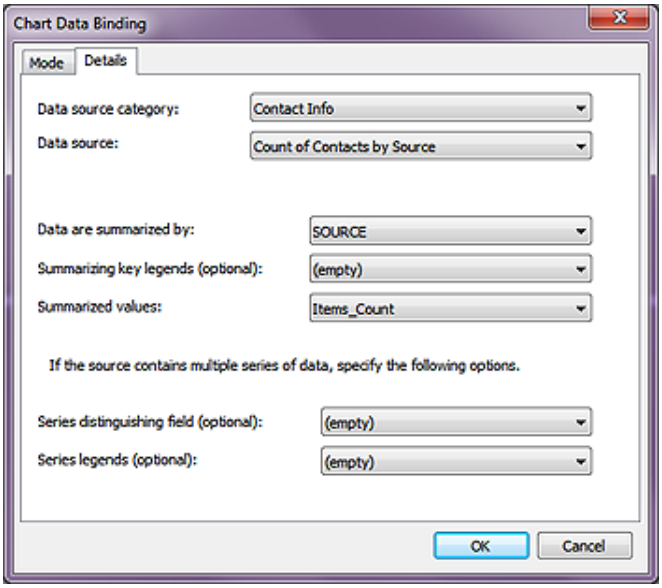


Figure 1-31

I certainly think so. Let's attempt to do just that right now by clicking on the **Events** button in the **Dashboard Design** mode to produce the dialog form as shown in Figure 1-32.

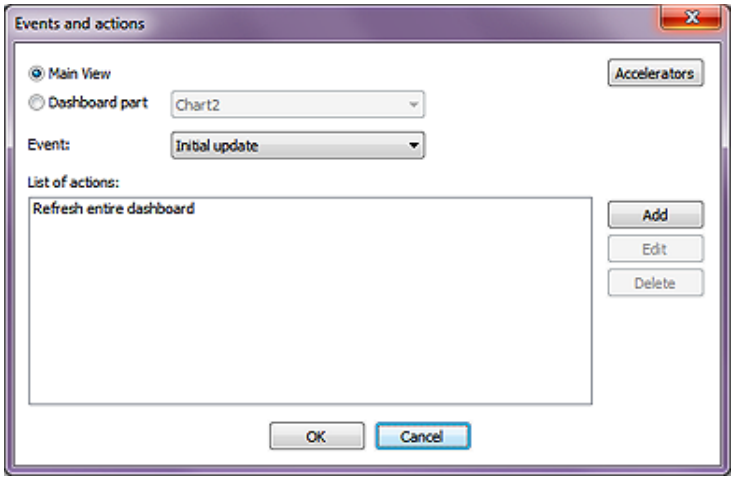


Figure 1-32

Events are the Triggers and Actions for components on the dashboards. First, one must select the component that this Trigger will function for, and then one must select the Trigger **Event:** that would cause this Trigger to be pulled. We know that this particular Trigger component is not the ◉ **Main View**, so we'll need to select the ○ **Dashboard part** which in this case is the **Count of Contacts by Source**.

We'll then have to supply a Trigger **Event:** for which to watch by clicking on the associated drop list and selecting an event. What's that you say, *"There is only one Event."*, and you'd be absolutely correct. The **Event:** is **State changed**, and will monitor this component for any changes in state like say a **MouseDown/MouseUp** (a mouse click) **State change**.

The **Accelerators** button disables the instant that one clicks on the ○ **Dashboard part** radio button, but would have permitted one to a hotkey against to ◉ **Main View** in case one wanted to pull the Trigger by issuing a hotkey stroke. More importantly, you have not added any actions in your **List of actions:**. That's right, you have multiple actions occur as the result of the Trigger being pulled. Because my screenshot was utilizing the GoldMine default settings there is already one action, **Refresh entire dashboard**, displayed in the list, however, as we selected the ○ **Dashboard part** radio button, we must now supply an action or actions to our empty list. I'll have you do just that by clicking on the **Add** button which brings forth the dialog form shown at the top of the next page in Figure 1-33. This is the same dialog form as I showed you back in Figure 1-13, however, I have modified this dialog form from its default state.

The **Action:** is **Drill Down** while the **Update dashboard part** is set to **Contacts by Source** which automatically picks up its **Data source:** of **Contacts Information with Source Field**. Then the **Runtime Parameter 1:** that you should use is the **Source**, and the ◉ **Runtime (from part)** remains as set by GoldMine at this point. Click on the **OK** button at this point, and then the **OK** button on the parent dialog form, and you should be finished with this section of the example.

If you have done everything properly (always a big if), you should be able to click on the **View** button at this point and test your dashboard out. When you click on one of the graph units, your table view should actually change to display just those records that were utilized in constructing that particular graph unit on your chart.

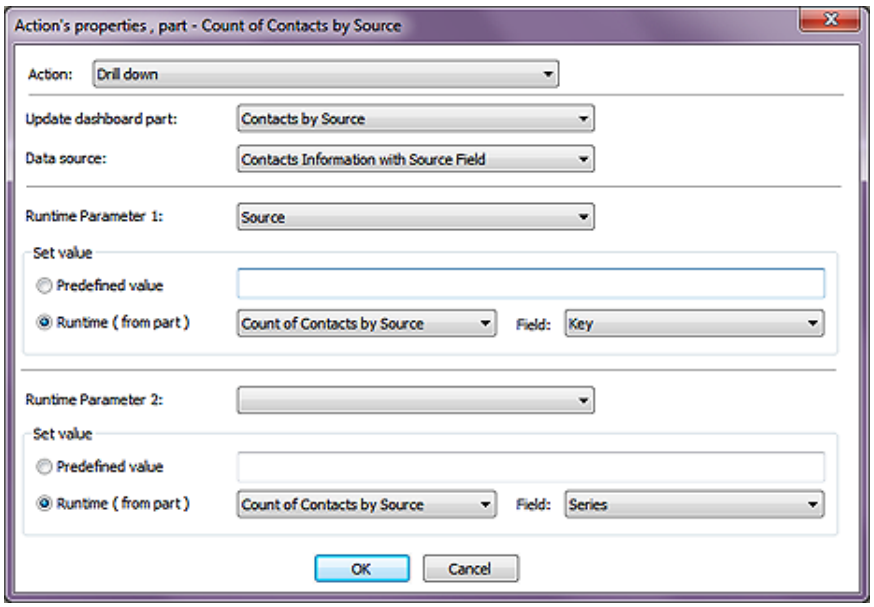


Figure 1-33

I'm thinking that yours is probably working properly at this point as mine certainly is.

Linking Dashboard Records to Contact Records

It's about time to go back and look at the linking capabilities of dashboards. GoldMine uses the **Contact1.AccountNo** field value for most of its own relationships between tables, although in some instances, GoldMine utilizes the **Contact1.RecID** field value. In order to set up a similar relationship in our dashboards, we need to capture (**select**) the **Contact1.AccountNo** field value. Please return to the **Contacts Information with Source Field** data source, and then modify the **Select** tab information by adding the **CONTACT(c1).ACCOUNTNO** field. We are done with the preparatory stage, so let's move on.

We need to make the **Contact1.AccountNo** field value part of our **Data Binding** now, so let's go ahead and **Edit** the **Data Binding** property as shown here in Figure 1-34.

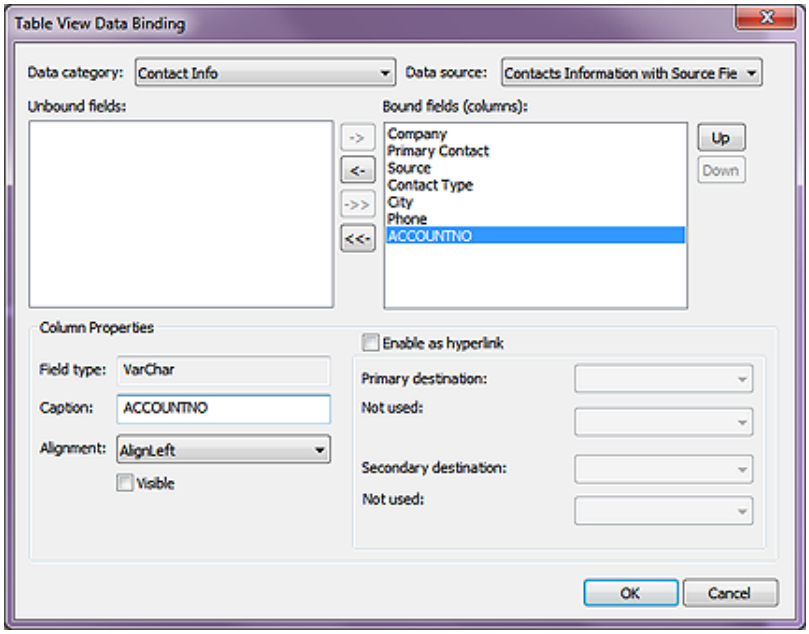


Figure 1-34

You'll notice that I have already pushed the **Contact1.AccountNo** from the **Unbound fields:** column to the last item in the **Bound fields (columns):** list, and as the information is meaningless to the enduser, I have selected to not make the column ☐ **Visible**. I'm not ready to have you leave the **Table View Data Binding** dialog form as yet, and we have more work to do.

Highlight the **Company** column in the **Bound fields (column):** list, and I'll have you work on that dialog form for that column next.

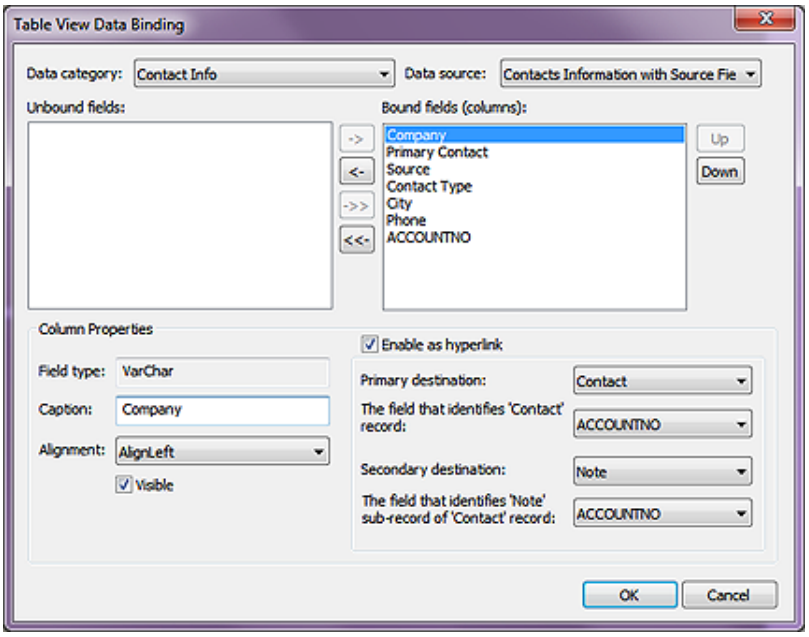


Figure 1-35

As shown here in Figure 1-35, I have already selected **Enable as hyperlink** for the **Company** column. Further, I have select the links **Primary destination:** to be the **Contact** while using the **ACCOUNTNO** as **The field that identifies 'Contact' record:**. Although, you probably don't need a **Secondary destination:** I have selected **Note**, and again **The field that identifies 'Contact' record:** will be the **ACCOUNTNO** field.

Note

If curiosity gets the better of you, and at this point it should, then I suggest that you examine all of the available destinations to which GoldMine is permitting one to link.

I do believe that we have set up all of the prerequisites at this point, hence, if you click on the **OK** button and then on the **View** button, you should see a change in your Table View now whether you click on a Chart unit or not. Does your **Company** column look similar to that which is shown here in Figure 1-36? It should. You should now be able to click on any **Company**, and the hotlink should take you directly to that record in your GoldMine. Additionally, when you arrive at that record, the **Notes** tab should be exposed if you chose the **Secondary destination:** as I had described in the previous paragraph. I do hope that you achieved your goal as well, and if not, you may want to revisit that in the book and try again.

Figure 1-36

Manually Typed Data Sources

To continue following Iains lead, we should discuss the **Manually Typed Data Source** now. The difference between the last data source that we created and the manually typed data source is that the former was produced utilizing a GoldMine Wizard to guide you through the various steps in creating a **New Data Source**. But you should know what goes into a data source at this point, and you should be able to create one manually if only you knew how to get into the proper area. You already know that the data source is nothing more than a bastardized SQL Query so let's continue with that understanding.

We'll mimic Iains videos again, and create a new Dashboard that is simple and that looks at Appointments by UserID over the last 60 days. So go ahead and click on the **New Dashboard...** hotlink, and let's get started. Immediately you'll want to set the **Properties** of **Category** to **Definitive Guide Examples**, and **Name** to **Definitive Guide Example 3**. Click on the **View** button to assure that your Dashboard appears under the correct tree branch, and then back again on the **Design** button to expose the **New Data Source...** hotlink.

Now click on the **New Data Source...** hotlink to bring us into the Wizard that we are so familiar with now, and in the **Type of data** drop list, I would have you select the **Manually typed** option. Let's leave the remainder of this screen alone, and click on the **Next >** button. Now would be a good time to complete the **Name:** value, maybe with something like **Appts last 60 days**. Don't forget to set the **Owner:** value to yourself, and click on the **Next >** button once more. We're not going to set any parameters at this time, hence, you would just click on the **Next >** button.

You now have the ability to set you SQL Query, and I'll have you type in the SQL Query as I've shown you at the top of the next page if you don't mind.

Tip

If you have more than a single database, and you wish to have this query specific to that database then your query should look similar to this one:

```
select UserID,
  count(*) as [Appointments]
from SQLGoldMine.dbo.ContHist
where sRecType = 'A'
  and OnDate >= getdate() - 60
group by UserID
order by Appointments desc
```

```
select UserID,
  count(*) as [Appointments]
from ContHist
where sRecType = 'A'
  and OnDate >= getdate() - 60
group by UserID
order by Appointments desc
```

You will learn more on SQL Queries later on in this book in Chapter 7, **Gathering the Data**. As well I'll show you more about the various table structures in Chapter 8, **The Tables**. You'll want to click on the **Check** button to assure that you enter the query properly, and if you receive the **Query is correct** screen then go ahead and click on the **Preview** button to validate that your query is producing the expected result cursor.

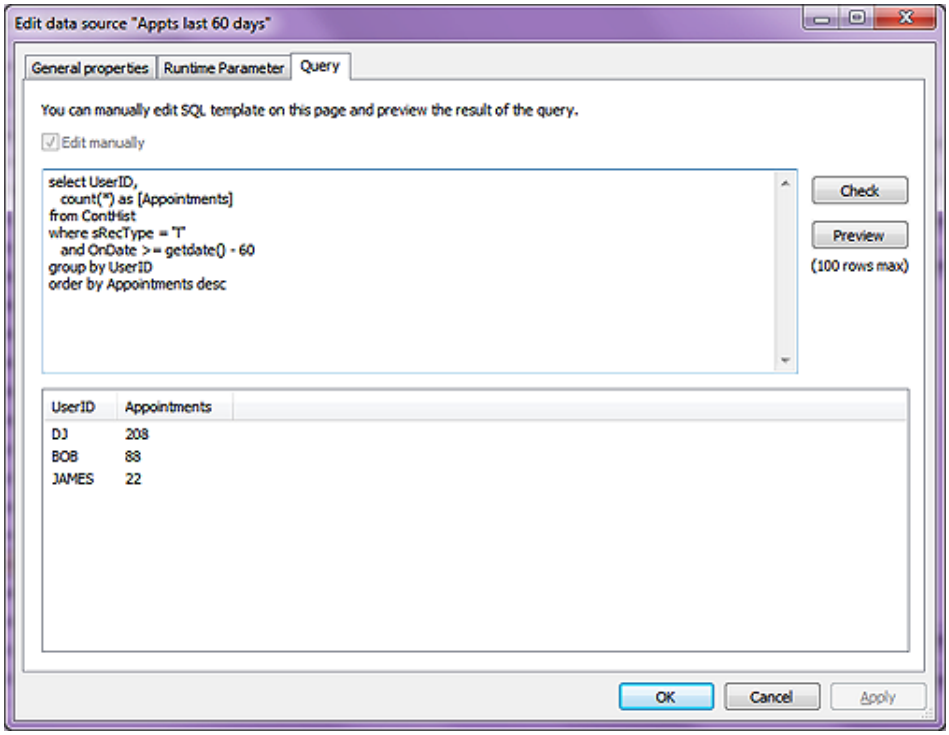


Figure 1-37

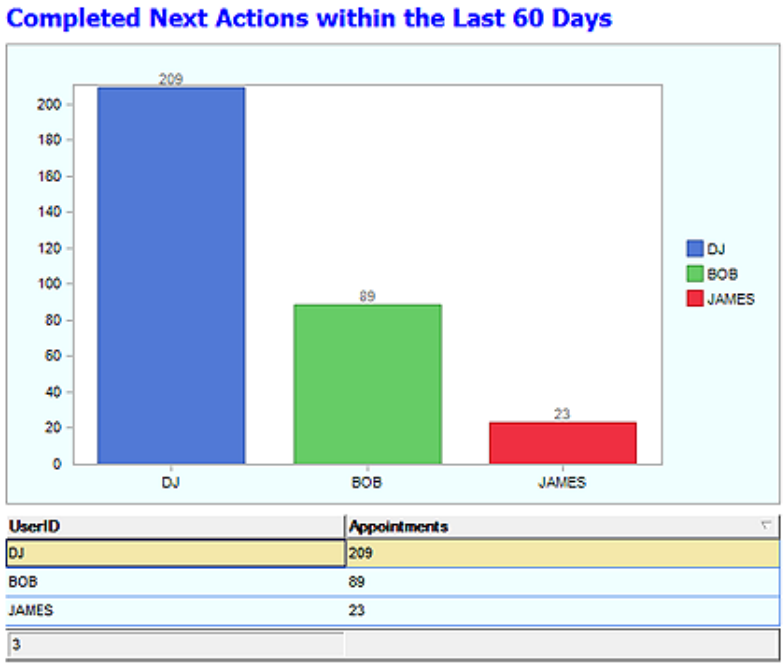


Figure 1-38

Does your Data Source dialog form look anything like mine as shown in Figure 1-37? The observant among you will notice that I have changed my SQL Query slightly as Computerese Inc does not really track Appointments as much as we track our Next Actions which are billable.

Alright then, go ahead and click on the **Finish** button to wrap up the creation of our manually entered data source. At this point you could easily add a Chart and/or a Table View to your dashboard, utilizing the **Appts in 60 Days** as your **Data Source**, and you've created another new Dashboard. You can see what I was able to accomplish using my Data Source in Figure 1-38 on the previous page.

Pivot Table

Iain has a good example of creating a **Pivot Table**, and I think that I want to follow along on this as he also incorporates another component, the **Drop-down List**, and utilizes that component to manipulate his **Pivot Table**.

Let's begin by creating a N**ew Dashboard...** utilizing the hotlink, and please throw this one into the same **Category** of **Definitive Guide Examples** while giving it a **Name** of **Definitive Guide Example 4**. You're an old hand at doing this by now I would think.

Use the hotlink, **New Data Source...**, to create, obviously, a new Data Source. This time we'll have you choose a data type of **Closed Sales**, and just start by selecting to ◉ **Start with a blank data source**. On to the next dialog form now shall we? Here we will supply a **Name:** of **Closed Sales by Quarter**, and please assign yourself as the **Owner:**. This time, as opposed to everything that we've done previously, we're going to ask you to select the ○ **Matrix** option. Pivot Table - Matrix, you see the relationship a bit later on.

Note

In the list, the functions are listed at the top, and are proper case while the fields are displayed below the function and are in all caps.

Moving forward to the next dialog form we have to set the **Base field:** value. For each Quarter we want to see the **Amount**, so we're going to ask you to select the **Amount** function from the drop list as we do not want an actual field here, but a calculation. The **Summary method:** naturally will be **SUM**. And now we come to the **Pivot field:** value. We're doing Closed Sales by Quarter so the logical pick has to be a by Quarter selection. We'd like you to pick **Activity fiscal quarter** from the functions in the drop list.

Next, we need to click on the **Add Pivot** button. In the **Add/Edit pivot custom column** dialog form, let's add a **Value:** of **1**, and a **Column name:** of **Q1**. Go ahead and add that, and then click on the **Add Pivot** button once more this time adding the **Value:** of **2**, and a **Column name:** of **Q2**. So you can see that we're adding a column for each of the Quarters in a fiscal year. Go ahead and add the last two Quarter columns on your own.

Now we want to add a **Users** column so we click on the **Add** button, and in the **Add/Edit field** dialog form select a **Field:** of **CONTHIST(h).USERID** which we know is a field as it is all caps. We won't need any **Aggregation:**, and the **Column name:** of **USERID** is popped in there automatically by GoldMine with our field selection. Please click on the **OK** button now.

For the **Predefined Search Criteria**, we'll accept the default as produced by our previous **Category** selection:

```
AND CONTHIST(h).SRECTYPE Equal to S
```

Now this doesn't hold true for the **Runtime Parameters**. We would like you to add a new parameter to the default supplied listing:

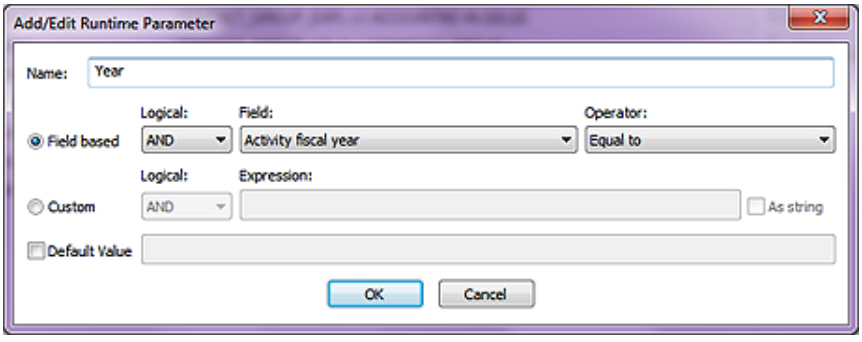


Figure 1-39

This will allow us to select, in the drop list, to affect the Pivot Table.

We'll leave the **Group by / Order by** dialog in its default state as we really have no need to change that for this exercise. You may want to **Check** your **Query** and/or **Preview** it. It should work just fine, and then you'll want to click on the **OK** button to save the **Data Source**.

This is the stage where we must begin designing our actual **Dashboard**. Why not start by dragging a **Table View** on to our palate, and stretching it to appropriate proportions. Not to feaster with it now as you can always adjust it later on. You will, however, want to supply a **Name:**, and I have used

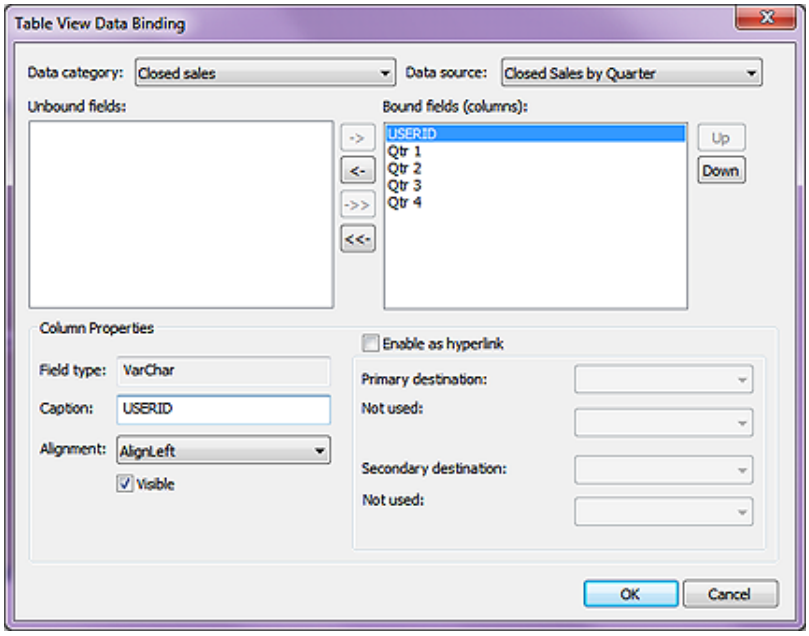


Figure 1-40

Pivot Table View for my test example. Additionally, you will want to do your **Data Binding** to your previously created **Data Source**. You can see above, in Figure 1-40, the **Table View Data Binding** dialog form from my test example. I did not select to **Enable as hyperlink** any of the fields that are bound. About the only other of the **Properties** that I changed was the **Layout Anchors** to **True, True, False, False**. That decision is strictly up to you however. I know that the curious among you, at this point, will want to see what see what you've accomplished so far, so go ahead and click on the **View** button, and when ready, click back again on the **Design** button.

One of the things that you will notice, if you clicked on the **View** button, is that you are viewing all Sales in each Quarter for all years. Now this is what we'll attempt to control via a **Drop-down List**. So go ahead and drag one of those components onto your palate, preferably above the previous created **Table View**. You may also want to add a **Label** component for your **Drop-down List** at this point as well.

We'll just give the **Label** component a **Text** value of **Year**. We'll also want to do a **Date Binding** for our **Drop-down List**. I would suggest that you use the **Manually Typed** for the **Data Category** with the predefined **Data Source** of **Years** which saves you from doing all of the hard work. Let's also change the **Text Field** value to simply **Text**, and the **Value Field** to **Value**. The **Value** will actually supply the current year from the **Drop-down List**. If you were to **View** your design at this point, you would find that everything functions properly, and probably looks great.

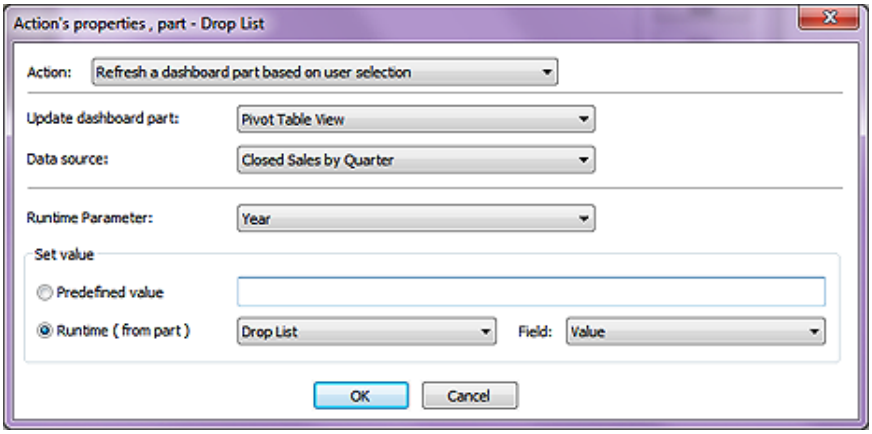


Figure 1-41

Ooops! Selecting a year from the drop list doesn't affect your table in any way now does it? I think that we have a little more work to do here. While in the **Design** mode, let's click on the **Events** button. For our ⦿ **Dashboard part**, we'll want to select your drop list by whatever name you gave it, while the actual **Event:** will be **State changed** as selected from the drop list. Please complete the **Action's properties, part Drop List** dialog form as shown on the previous page in Figure 1-41.

Click on the **OK** button, the **OK** button again, and then the **View** button. What do you think? Does your **Dashboard** function as expected now? I'm thinking that it does. You can spend some more time on this prettying up for the general populace, but for this book I think that we are finished with our **Pivot Table Dashboard**.

To keep this book a reasonable size, I must stop the **Dashboard** chapter here. Iain, however, has a few more videos on Dashboards that accompany this book. You may want to review those in your own time. My time, however, is finished, at least in this chapter.

2

In This Chapter

Tip

Read each section through, in its entirety, once, before attempting any changes on your GoldMine Premium. Then review the section as you are making the changes to your GoldMine Premium.

Note

Henceforth, in this chapter, the GoldMine Premium product will be addressed simply as GoldMine.

Note

*Feel free to add as many UserIDs as you would like. You are not limited by the number of users that you can have under the **Users' Master File**. You are only limited by the number of users that can log into GoldMine concurrently, and this is controlled by your purchased licenses.*

Tip

*The GoldMine Administrator can access the **UserID Properties** dialog form for a given user by simply double-clicking anywhere on the users record in the **Users' Master File** dialog form.*

Users' Settings

The GoldMine Premium Edition line has been available to GoldMine users for over a year as of the date of this writing. This is the second edition of **The Hackers Guide to GoldMine Premium**, and this book is being rewritten primarily to accommodate the changes that have been incorporated into the GoldMine Premium 8.1 product.

Users' and User Groups are where much of the Security in GoldMine Premium will be established. I will do a detailed walk through of this section of the GoldMine Premium application. From within the **Users' Master File** dialog form, the GoldMine Premium Administrator will pre-define the UserID.ini files, and configure/control the rights that a user has, while using the GoldMine Premium application. Within the **User Groups Setup** dialog form, the GoldMine Premium Administrator will establish user clusters that could be applied to further control security in the GoldMine Premium application in areas such as Record Ownership or Field Level Record Typing. The **Resources' Master File** dialog form is where the GoldMine Premium Administrator will configure their corporate resources if they so choose. The **GoldMine License Manager** is the area where the GoldMine Premium Administrator will manage license distribution as well as maintaining the GoldMine Premium Registration information. They may also update their GoldMine Premium from here, as newer builds/versions become available. Typically, your GoldMine Premium may be updated to newer builds of the same version regardless of whether you have an active Maintenance Plan with FrontRange or not. Typically, your GoldMine Premium version may be updated to newer versions of GoldMine Premium, however, this should only be possible if you are currently active on a Maintenance Plan with FrontRange Solutions.

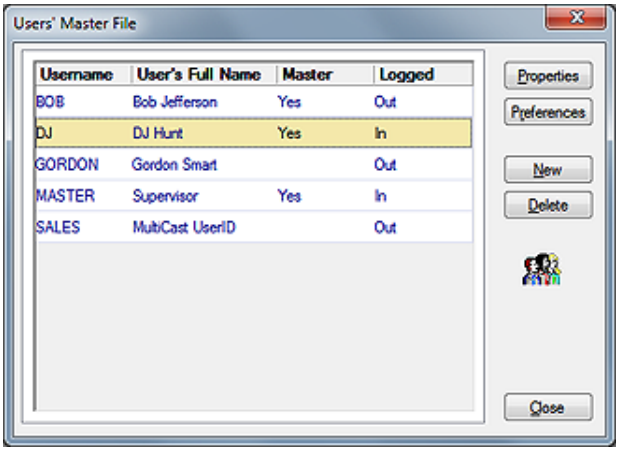


Figure 2-1

Let's start right in, shall we, with the **Users' Master File** which is accessed from the GoldMine main menu under **Tools** | **Users' Settings...**, and will bring up the dialog form shown above in Figure 2-1. By the way, this is the first security feature, as this menu choice will not even be available unless the user trying to access the option has **Master Rights** within GoldMine. In Figure 2-1, the reader can see five buttons, and they are:

Properties
Preferences
New
Delete
Close

As well, on this dialog form are displayed all of the users that have been set up to date to use your GoldMine installation. From this dialog form the GoldMine Administrator can view some basic information about each user. There is a column for the **Username**, the **User's Full Name**, whether or not the user has **Master Rights** (Yes or blank if No), and finally, whether the GoldMine application believes that they are currently Logged In or Out. From this dialog form, the Administrator can view some basic information about each user.

The **Properties** button will allow the GoldMine Administrator to edit the properties of the currently highlighted user. This dialog form is the same dialog form, that results from the clicking of the **New** button, the opening page of which is shown in Figure 2-3. I'll discuss the properties based upon the **New User Properties** dialog form in a moment.

Before I actually begin discussing the properties assigned to a user, I would like to briefly mention the **Preferences** button which, in other areas of GoldMine Premium, is also known as the users **Options**. The Preferences button allows the GoldMine Administrator to access, and preset the networked individuals GoldMine User Preferences/Options. As these are user preferences/options, they can be modified at any time, by any user. The user preferences/options will be discussed in detail in Chapter 3, however, having the ability to adjust the users preferences/options here, just allows the GoldMine Administrator to apply any corporate standards that have been established for new GoldMine users. Personally, I would prefer to set my corporate standard through the use of the GM.ini, and its Corporate Override capabilty which I will discuss later in Chapter 3.

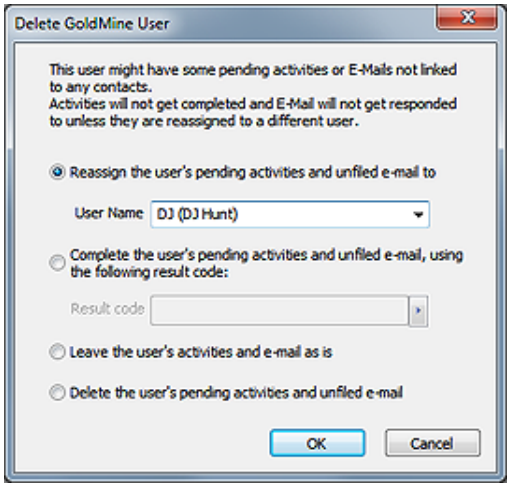


Figure 2-2

The **Delete** button, in Figure 2-1, does just what it states. Clicking this button will delete the highlighted user, but not until you select an option. This is a total revision in GoldMine Premium since the last edition of this book was released. Hence, we have a lot more to discuss in this edition of the book about the **Delete** button.

I would like to quote the explanatory dialog on this form:

"This user might have some pending activities or E-Mails not linked to any contacts.

Activities will not get completed and E-Mail will not get responded to unless they are reassigned to a different user."

FrontRange has added this dialog form in its attempt to reduce the orphaning of records within the GoldMine database. You will notice that you have four radial button options from which you may choose. The first of which being the option that is selected in the default state:

- ◉ **Reassign the user's pending activities and unfiled e-mail to**

  User Name DJ (DJ Hunt)

Keeping this option will assign all of the deleted UserIDs Pending Activities, and any Unfiled E-mail Messages, which are also Pending Activities for the deleted UserID, to a new UserID selected by you from the list of remaining UserIDs. Choosing this option willl allow another UserID to handle, or not, the activities that were assigned to, and should have been handled by the deleted UserID.

- ○ **Complete the user's pending activities and unfiled e-mail, using the following result code:**

  Result code COM

Utilizing this option, will cause all of the deleted UserIDs activities/unfiled e-mails to be Completed. No one will have a chance to review these to see if they should have been Completed, and the records will just be relegated to history.

And finally we have the last two options which I will list here, and explained below.

- ○ **Leave the user's activities and e-mail as is**
- ○ **Delete the user's pending activity to an unfiled e-mail**

To leave the uses activities and unfiled e-mails as is, is to abandon them and leave them at the orphaned records. Obviously this is not a good option, and in my opinion was unwise for FrontRange to add as an option.

On the other hand, to delete the users activities and unfiled e-mails, would at the very least, take care of any possible orphaning of records, however, there is always the possibility that some of these user activities in unfiled e-mails should have been addressed. I think that I would be cautious of using this option as well.

In the **New User Properties** dialog form, shown on the next page in Figure 2-3, the first field is the **Username:** field. The GoldMine application forces capitalization of the information contained in

this field, and the field will accept up to eight characters (refer to sidebar Note).

The next field is the **Full Name:** field, which is the name that GoldMine relates with this UserID. In this field you may enter up to 26 characters representing this UserIDs full name. GoldMine will pull the information from this field wherever the user employs the **&UserFullName** macro (refer to Appendix B). This macro will most commonly be employed in e-mail templates, document templates, and reports, and the users may find it useful in many other areas. It is for this reason that you would want to employ a corporate standard for the presentation of this information.

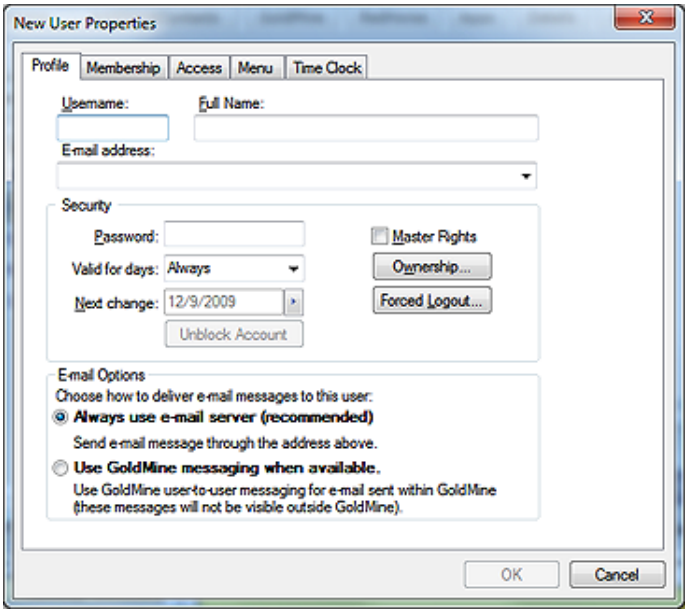


Figure 2-3

New in GoldMine Premium v9 is the **E-mail address:** field. You are encouraged to include a UserID E-mail Address for this new user. You will understand the logic for this better when we discuss the **Email Options** frame which is also new in this dialog form.

Next, on the **New User Properties** dialog form, is a frame for the **Security** settings. At this point, the GoldMine Administrator may want to fill in the **Password:** field with a temporary password while creating this UserID in GoldMine. The user will have the chance to change their password at any time through their own **Options** dialog form. Personally, I do not see any requirement to assign any user a password with which they would need to use to log into the network GoldMine except for those users possessing Master Rights, or in a cut throat environment. On the other hand, I would require all of my remote users to have a password on their GoldMine. All users will have access to any other users Calendars, Groups, Filters, etcetera, unless these rights are specifically removed from the user. Applying a password to the user name for login, in most situations, is for the users perception of security, and not for actual data security.

If, and where, passwords are employed, the period of time for which they are valid should be specified. GoldMine supplies the **Valid for days:** field in which the GoldMine Administrator may select a number of days from the drop-down list (**Always** being the default state). Even though you may type into this field, only information validated from the drop-down list will be saved into the field. This list is not user definable within GoldMine. Once any numeric value is entered into this field, the **Next change:** date field, will immediately be enabled.

The date, in the **Next change:** field, defaults to the creation date of the UserID within GoldMine. The GoldMine Administrator creating this user identity may set the date that the user must first change their password. All required changes to the password, thereafter, will be based on the **Valid for days:** entry, and this will be counted from the date that the user actually makes the change to their password.

New in v9 is the disabled **Unblock Account** button. If a user fails to login after 3 attemps, the account becomes disabled in GoldMine and can only be reinstated by a user with Master rights. To enabled the user's account once it has been disabled, simply click on this button, and then on the **OK** button.

The next option, to the right, in this frame that I want to discuss, and the single most important setting in GoldMine, is whether to allow this user to have ☐ **Master Rights** which, by default, is not selected. Allowing any user Master Rights is to allow them full access anywhere in GoldMine. A person given Master Rights can see everything, and I mean absolutely everything within GoldMine. If an e-mail is marked as Private, a user having Master Rights can see and read it whether it is theirs or not. Usually Master Rights are reserved for the GoldMine Administor(s). I have seen many companies distribute Master Rights to many users in the company, and then have the audacity to turn around and wonder why the GoldMine Security features do not work properly. As my wife might say: "Go Figure".

Note

Whereas, in prior versions of GoldMine, you could use special characters in the UserID to assist with the sorting of the UserID's in the various lists, in this version of GoldMine, you cannot use special characters when creating the UserID via the GoldMine GUI. You may, however, still use special characters when adding the UserID programmatically.

Recommendation

I do not feel that passwords are required in the GoldMine application. At best, they just prevent other GoldMine users from logging in under someone elses UserID.

*It is my recommendation that you not use GoldMine passwords except for those users who have Master Rights. The GoldMine Administrator can access the **UserID Properties** dialog form for a given user by simply double-clicking anywhere on the users record in the **Users' Master File** dialog form.*

WARNING

GoldMine on Remote notebooks should be required to have a login password. If you will be synchronizing the Users file to the Remote-side notebooks, then the Remote-side UserIDs should have a login password.

Recommendation

*I feel that only two individuals in any organization should have **Master Rights**, the Primary GoldMine Administrator, and a backup GoldMine Administrator.*

*As a consultant, many times, while working on a clients GoldMine, I will often enter a UserID of **DJ** possessing Master Rights into their GoldMine UserID list as a back door in case their GoldMine Administrator forgets their password or leaves the organization which happens more frequently than you may think.*

Note

I can't remind you enough that UserIDs possessing Master Rights trump all GoldMine Security settings.

Figure 2-4

You next encounter the **Ownership** button. I will discuss this now as it is present in this dialog form in the Security frame which I am currently discussing, however, it will be more meaningful after the GoldMine Administrator has defined User Groups. Clicking on the **Ownership** button will bring up the dialog form shown here in Figure 2-4. What I am defining here is the default ownership for any new contact record that may be created by this UserID within GoldMine. If a new record is created with ownership, then no one can edit the information contained on portions of the record unless they are the Owner, or a member of the User Group that owns the record, or unless they have Master Rights (trump). The default **New records owner:** is the User Group (public). The GoldMine Administrator may change this to any existing UserID, or existing User Group by selecting that item from the drop-list. The GoldMine Administrator may choose to change this later, after they have had a chance to setup the User Groups.

In addition, at this time the GoldMine Administrator may establish Record Curtaining. The default setting is ◉ **None**, and means that all users will be able to see the information contained in the contact record while not necessarily having the ability to edit the information in some areas of the record. An administrator may select a UserID default curtaining of ○ **Semi-partial (hide tabs)** which blanks the display of the tabs area, however, there is no longer any curtain displayed as us old time GoldMine users had been used to seeing. Selecting a ○ **Partial** curtain, on the other hand, will blank out most of the contact record for all users that do not own or are not part of the user group that owns the record. Only the top few rows of the contact record will be visible to non-owners. Selecting ○ **Complete** will hide the record completely for any of those UserID's that are not the Owner or part of the User Group that is the Owner. Where, in past versions of GoldMine, a non Owner could schedule an activity against a fully curtained record, in GoldMine Premium they will no longer have this ability.

Recommendation

I feel strongly that, as a GoldMine Administrator, you should employ the Forced log out feature as it will free up idle licenses, and may log out a user who left GoldMine running inadvertently. Most backup utilities will not backup through open files, hence, you would want to do everything possible to assure that GoldMine is not left running on any workstation.

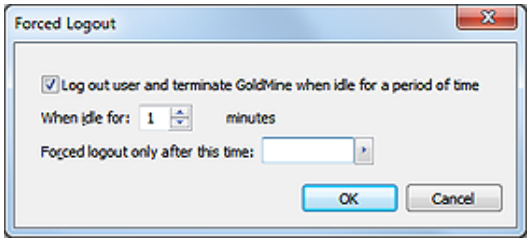


Figure 2-5

In v9, we next encounter the **Forced Logout...** button as depicted in Figure 2-3 on the previous page. Clicking on this button results in the dialog form shown here in Figure 2-5. The **Forced Logout** is often overlooked by GoldMine Administrators, yet I feel that it should play an important role in your organization, and that its use should be considered. This is not the perfect mechanism by any means, but it does work. Most backup programs will not allow the backing up of open files. This dialog form allows the GoldMine Administrator to force the user out of GoldMine when there is no activity within GoldMine for a specified amount of time. Once the GoldMine Administrator selects the ☑ **Log out user and terminate GoldMine when idle for a period of time** option, the rest of the dialog form is enabled. From there, it is only necessary to set a **When idle for: x minutes** setting, however, I normally suggest that the **Forced logout only after this time:** field be set to around **6:00 pm**. In most offices, most users should have finished work by that time of the day, and, again, they will not be forced out at that time if they are actively utilizing GoldMine. I usually set the **When idle for:** setting to **1 minutes**. Oh, well, I'm taking this right off of the dialog form, don't blame me for the grammar. Anyway, with these settings, you are instructing GoldMine to shut itself down if it is after 6:00 pm, and if there has been no keyboard/mouse activity within GoldMine for that UserID for at least 1 minute. As the GoldMine Administrator, you may want to tweak these settings to better meet your corporate needs. At the least, with this, you have somewhere to start.

v9 also provides us with more new options in this area. We now have a frame labeled **E-mail Options** which includes one choice as a two radio button selection. This choice will determine the requirement for or not the need for an **E-mail address:** for this UserID. The preface for your selection reads as follows: **Choose how to deliver e-mail messages to this user:**, and one of the options is clearly marked as recommended.

- ◉ **Always use e-mail server (recommended)**
  Send e-mail message through the address above
- ○ **Use GoldMine messaging when available.**
  Use GoldMine user-to-user messaging for e-mail sent within GoldMine
  (these messages will not be visible outside of GoldMine).

After years of my advising you to establish a GoldMine record for each of your GoldMine UserIDs. After years of my advising you to use Internet E-mail to communicate with your users as opposed to the GoldMine internal messaging system, well, FrontRange has now rubber stamped my advise with the addition of this section to the **New User Properties**.

Recommendation

Workstation/Remote

Contact Record settings:

☑ ***Add New***
☑ ***Edit Fields***
☑ ***Edit Tab Folders***
☑ ***Schedule Process***

General Access settings:

☑ ***Build Groups***
☑ ***Issue SQL Queries***

Access to Others settings:

☑ ***Calendars*** *(all)*
☑ ***History*** *(all)*
☑ ***Forecast*** *(Specific User Group)*
☑ ***Reports*** *(all)*
☑ ***Template*** *(all)*
☑ ***Filters*** *(all)*
☑ ***Groups*** *(all)*
☑ ***Links*** *(all)*
☑ ***Opp/projects*** *(all)*
☑ ***Cases*** *(all)*

I will skip right over the **Membership** tab, and continue on with the **Access** tab. The **Membership** tab deals with **User Group** membership which I will be discussing later in this chapter in the User Groups section. Under the **Access** tab, shown here in Figure 2-6, the GoldMine Administrator is offered four frames under which they may control the access rights for a given UserID. The **Contact Record** frame, the **General Access** frame, the **Notes Access** frame, and the **Access to Others** frame. These are the four frames in which you will define a users access rights.

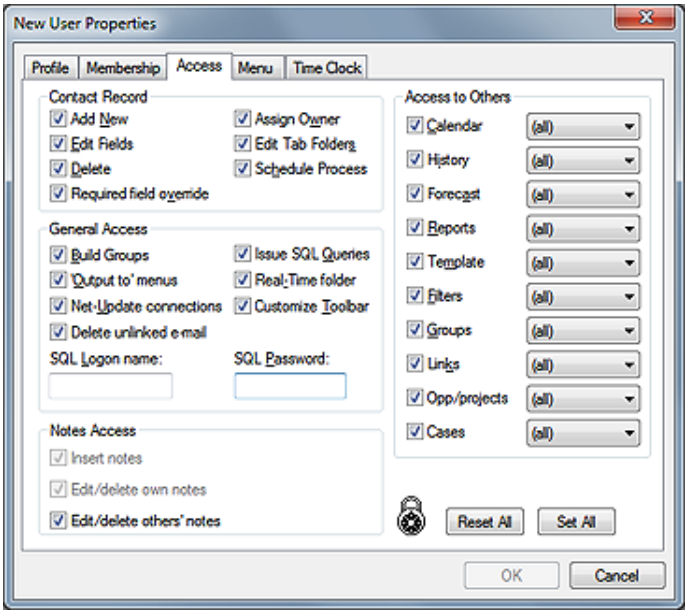


Figure 2-6

I begin by looking at the **Contact Record** frame. This frame contains seven option switches which the GoldMine Administrator may turn Off (☐) or On (☑). They are all on in the default configuration shown above.

The first option allows the UserID to ☑ **Add New** contact records to the GoldMine database. You may have assigned the adding of records to a specific user, or to a group of users, in your organization for consistency of data input, and you may **not** want certain individuals to have the ability to add contact records. If that is the case, then, for those individuals UserID, you would uncheck this option.

I discussed another way of accomplishing the same end result as this next option does when I previously discussed **Ownership**. The ☑ **Edit Fields** option means that this UserID is allowed to Add, Modify or Delete field data within GoldMine. Remove the check from this option, and the user will not be able to modify field data, even if they are the owners of the record. I want to emphasize the word **field**. This means any field on the screen, those that are contained in the **Contact1**/**Contact2** tables at least, are or are not editible as determined by this setting. This would be any field in the top portion of the main GoldMine screen including the **E-mail:** and **Web Site:** fields (Yes, I know that these are not really fields, nor are they in the Contact1/Contact2 table), as well as any field under the **Summary**, the **Fields** and the **Notes** tabs of the main GoldMine screen. You should be aware that the user will still be able to add E-mail Addresses and Web Sites, but it must be done under the **Details** tab, and not from the displayed field information.

Recommendation

Designate one field to be marked when a user wants a particular record to be deleted from the database. Then assign a Master Rights UserID to review these daily, and to determine if they truly should be deleted.

Upon a determination that a record should be deleted, the UserID possessing Master Rights would archive (move) the record to another GoldMine database. I never recommend deleting a record once it is contained within my database.

For those records that the Master Rights UserID determines should ***not*** *be deleted, it would be incumbent upon them to remove the deletion marker at the end of their session.*

WARNING

In order for the ☑***Required field override*** *to function properly, GoldMine requires that each and every UserID Login have a Login Password.*

The ☑ **Delete** option allows this UserID to delete contact records from the GoldMine database on an individual record basis. Only users possessing Master Rights have the ability to perform en massé deleting of the contact records in the database. If your organization wants to assign the right to delete individual records to a select group of individuals, then this is where you would do that. Remove the check, and you remove this UserIDs ability to delete contact records from the GoldMine database. This time I want to emphasize **contact records**. The user will still maintain the right to delete pending and historical information. Should you have a group of users that you have removed their ability to delete records, then you should provide them with another way of flagging a record for deletion. This way, those that have deletion rights will be able to group these records, and will be able to delete the records en massé that they have determined should actually be removed.

There is an option that has been available since GoldMine 6.6 for ☑ **Required field override**, and although I'd played with it for a while, I could not get it to function as specified. From the GoldMine Help files: *"**Required field override:** Authorize an override on a required field"*. From my previous testing, any user can select any other user and override any required field. I have retested this in the current Beta build of GoldMine Premium v9, and it still does now appear to function as expected.

The next option, over which you have control, is whether the user may or may not ☑ **Assign Owner** to a record. I had discussed ownership earlier in this chapter. If you have assigned ownership based on the UserID creating the contact record, then this option is probably redundant, and any changes to ownership should be accomplished by a GoldMine Administrator. Alternatively, changes in ownership may be controlled via your Lookup.ini (discussed elsewhere in this book). If you are not assigning ownership at the time of record creation, and based on the UserID, then, possibly you would not want to leave this right to your user(s). See my recommendation in the sidebar on this page for Workstation/Remote users as a possible beginning point for your configuration.

The ☑ **Edit Tab Folders** option allows the user to create new, modify existing, and delete records under the **Contacts**, **Detail**, **Referrals**, **History**, **Links**, and other tabs. Remove this option, and they will not be able to create, modify or delete records under these tabs. If this option is removed for the user, they will still be able to **Complete**/**Schedule** activities under the **Pending** tab, but they will not be allowed to modify or delete any existing activity there under the tab, or anywhere that the activities are being displayed. Activities are displayed on the **Graphical Calendar**, and in the **Activity List** as well as under the previously discussed **Pending**/**History** tabs.

The last option in the **Contact Record** frame is whether the user is allowed to ☑ **Schedule Process**. This refers to the users ability to attach a GoldMine **Automated Process**, and not to an actual scheduling of an activity. In the best set GoldMine environments, each record automatically has one automated process attached at the time of record creation (refer to the chapter on Automated Processes the Observer Track). This process then watches for various activities, or actions, and, in turn, attaches any additionally required Automated Process Tracks to the contact record in question. The user is rarely required to attach an Automated Process Track, and this whole activity might be best left to the GoldMine Administrator. Possibly you could develop a **S**tandard **O**perating **P**rocedure (**SOP**) by which this process could be escalated to an authorized individual.

Note

Readers of the previous versions of my books, The Hacker's Guide to GoldMine Premium, will notice that this frame has changed in this edition of the book, and build of GoldMine Premium.

The next grouping that I will handle, and as was shown in Figure 2-6 on the previous page, is the **General Access** frame. The rights in this frame are not contact record specific, but instead, apply to GoldMine as a whole. As before, the user begins with all these rights set in the default state, and it is up to the GoldMine Administrator to determine if this UserID should retain these rights.

In the first option, you must decide is if you want this UserID to have the ability, or not, to ☑ **Build Groups** in GoldMine. In this case, I am talking about the ability to build Contact Record Groups, and this should not be confused with the GoldMine Administrators right to create User Groups.

The next option that you have for the UserID is, do you want them to have the access to the ☑ **'Output to' menus** item of the local menus. If an organization is really concerned about their contact information being taken outside of the corporation, then this may be an option that one would be likely to take away from their everyday GoldMine user. As I explain in a later chapter, the **Output to ►** option, from the local menu, will allow the user to output data to the Printer, Word, Excel, or to the Clipboard. I discuss the powerful advantages of this ability under the **SQL Queries** section in the **Gathering the Data** chapter of this book, while I discuss its disadvantages here. As the GoldMine Administrator, you will need to weigh the advantages against the disadvantages, and your corporations security needs to determine whether this option should remain selected for the everyday GoldMine user. Taking away this right for the user may be unnecessary, however it does erect another barrier that one must overcome to abscond with your data. Using the Microsoft SQL back end, however, affords you the greatest opportunity for keeping your GoldMine data in your hands alone.

Note

*As of this book, GoldMine Premium - The Definitive Guide, I am no longer recommending the use of the **Firebird** back end. In fact, just the opposite. I am only recommending the use of the **Microsoft SQL Server** back end.*

Between the backend and the GoldMine access rights, you should have no trouble securing your data. If this is your goal, you should not use general SQL access to your database, but, instead you should use SQL Secure Logins.

The next option is the option that I alluded to previously. I have found that most organizations will uncheck this option for their everyday GoldMine user. The ☑ **Net-Update connections** in a network environment is not required, and could cause problems if not performed by the GoldMine Administrator, as well, a Net-Update can not be accomplished against an Undocked or Site License. Only one person, the GoldMine Administrator, should be capable of updating the GoldMine application to the latest version or build. This right should not be given, in most cases, to the individual user. There are conditions where this option might remain selected, yet none come to mind at the moment. In most cases, however, this option should not be selected.

GoldMine Premium, while removing some of the previous **General Access** options that we were used to having, had added a much needed option which is the ability to ☑ **Delete unlinked e-mail**. Haven't we longed for this option, and it is selected by default also. Do you remember in GoldMine past when you had to Read/File an e-mail of this type before you were permitted to delete it? Well, this is no longer the case. I think that you'll learn to appreciate this option, and will opt not to deselect it for any of your users. Yet, that is your decision to make, but don't blame me if you do and you receive tons of complaints from your endusers.

Now you are asked if you want this UserID to have the ability to ☑ **Issue SQL Queries**. All of these user rights settings go hand in hand. You really wouldn't need to worry as much about the previously mentioned Output to option, if you took away the users right to issue SQL Queries. Once you have made the decision for one, then you would probably apply the same rule to all of these options. I believe that the SQL Query tool is probably the single most important data mining tool within your GoldMine solution, although, Universal Search is right up there in importance.

WARNING

*As of this build, GoldMine Premium 9.0.0.55, the turning off of the ☐ **Real-Time folder** has no effect at all via the Activity List. A non Master Rights UserID simply needs to change the UserID to a UserID that does have these rights, and the Real Time Activities will be displayed for that UserID.*

Tip

*In my opinion, one of the main reasons for using the SQL backend is for its security features. To use the **sa** (System Administrator) account is tantamount to circumventing the advantages of the SQL security.*

It is my feeling that each user should be set up with a SQL identity, and have their own password.

As an individual, however, I have been known to personally ignore my own advise.

Note

*Whereas the Notes used to be stored in a Text based column of the **Contact1** table, as of the 8.5 build of GoldMine Premium, the Notes are stored in a table of their own, conspicuously named **Notes**.*

The ☑ **Real-Time folder** should probably not be selected. The Real-Time tab is viewed from the **Activity List** dialog tab, and selected from the Navigation Pane tree. This tab view displays all of the completed activities for all of the users in the order in which they were completed. Many GoldMine Administrators had been asking for the ability to remove this tab from their users view for many years. The FrontRange organization has supplied this option to fulfill that purpose. Now that it is available as an option, I find that most GoldMine Administrators are unchecking this option. When this option is not checked, the **Real-Time** option will still display on the Navigation Pane tree, yet it will not be enabled. Remember, Master Rights for a UserID trumps all, including the **Real-Time** option setting.

The ☑ **Customize Toolbar** allows the users ability to Customize their Toolbar. If you have a need for a Corporate Standard Toolbar, then you may want to unselect this right for your non Master Rights users.

The last two fields in the **General Access** frame pertain to the SQL Login/Password, and how the SQL databases were setup for that environment. If the SQL environment was setup with the default username **sa** (System Administrator), with no password (not recommended), then these two fields should remain blank. If, on the other hand, your SQL environment has individual security accounts (recommended), then you would want to include the users **SQL Logon name:** and **SQL Password:** in the respective fields. This will permit the user to bypass the SQL Server login dialog form when entering GoldMine Premium, and they are not utilizing the aliased login. The **SQL Logon name:** and **SQL Password:** fields will each accept, what appears to be, unlimited alphanumeric characters, at least in GoldMine Premium 9.0.0.55. Obviously, I think, the password field will only display asterisks (*) or bullets (•) depending on your OS, and will never display the actual password nor will the actual password be saved unencrypted.

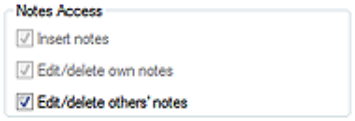


Figure 2-7

In the next frame, the **Notes Access**, which is new to this build of GoldMine Premium, we now have the ability to control a UserIDs capabilities to work with Notes. I do have to explain this functionality a little as these are check box options with an intelligent relationship. The default condition is displayed here in Figure 2-7. Please notice that the first two options are selected, albeit, disabled, while the third option is selected, but enabled. If one were to remove the UserIDs ability to ☑ **Edit/delete others' notes** then the ☑ **Edit/delete own notes** would then become enabled. Finally, if one removes the UserIDs ability to ☑ **Edit/delete own notes** then the ☑ **Insert notes** option is enable, and you must decide whether to permit this UserID to insert notes into GoldMine at all. So we now have a cascading selection for the control of the users capabilities under the **Notes** tab.

In the last frame, the **Access to Others** frame, all of the options function the same, and selecting or deselecting will turn on, or off, access to others ☑ **Calendar**, ☑ **History**, ☑ **Forecast**, ☑ **Reports**, ☑ **Template**, ☑ **Filters,** ☑ **Groups**, ☑ **Links**, ☑ **Opp/projects**, or ☑ **Cases**. The latter two being new additions to GoldMine Premium as of 8.1.0.40. Also new in that build, and not as noticeable, is that the order of the options has been changed. In fact, I did not even notice the change until I started updating this chapter.

In addition, for those options that are selected, you may further qualify to which User Group this option, for this UserID, pertains. You may set, for this UserID, that they have the right to access others ☑ **Forecast** (not recommended) sales, but only if those forecasted sales belong to members of the (sales) User Group. I will be discussing User Groups later in this chapter, however, this is one of many instances where the User Groups could be employed within the GoldMine environment.

I want to move on now to the **Menu** tab, as shown on the next page in Figure 2-8. From this tab, you can define what will appear or function in the users Main GoldMine menu, and, of course, what you do not want to appear or function there as well. Some of the menu items, though listed in this area as being definable, will not even show on the users menu if that user does not have Master Rights. For instance, though it shows up as being definable, the **Tools** | **Users' Settings...** menu item will not show on the users menu, regardless of the setting established here, if they do not possess Master Rights.

One of the important items on the **Menu** tab is the ability to **Save** a menu, once defined, as well as **Delete** a defined menu. If you have a standard UserID defined with menu options, and you are cloning that user to create new standard users, then the ability to save menus in different configurations may not be as useful to you. If, on the other hand, you are creating each user individually, you may wish to create positional menus. For example, you might create a menu for the manager positions, another for the secretarial positions, and yet another for any other positions in your organization. Then you would only need to select the menu for the user position that you are defining, instead of having to create a menu for each and every user individually.

The actual act of defining a menu is very simple. I suggest that you begin by clicking on the **Expand** button. This will expand all levels of the menu tree to its fully opened tree menu structure. In the

WARNING

I recently had two separate organizations screw down their security so tight that they removed the Tools | Users' Settings... menu item from even their Master Rights users, and no one was then able to add/remove GoldMine UserIDs.

Sometimes security conscious people scare me. They are so security conscious that they forget to think through the ramifications of their actions.

This cost them big dollars to have me rectify the situation, and cost one of them their job.

Hence my caveat: Read the book through thoroughly before doing anything, and then think twice about what you plan to do.

Note

*The old **Resize** menu option has gone the way of the Dodo bird, and you can no longer display user defined views as menu choices as you could in the past using this trick.*

WARNING

Regardless of what you've read or heard elsewhere, you want to make sure that all of your databases are backed up nightly on a rotating schedule. Whatever schedule you employ for your other office software should include your GoldMine folder, subfolders and databases.

Failing to do so may result in unrecoverable data loss.

You should never use, nor rely on the GoldMine Internal Backup/Restore features. They simply can not be reliabled upon.

Note

*The **"Big Brother"** feature may be turned off corporately via the **GM.ini** as follows:*

[GoldMine]
AutoLog=0
;AutoLog=1

*The semi-colon (**;**) preceeding an ini statement **Remarks** out that statement, and that statement will be ignored when the ini is processed. I use the semi-colon a lot to add comments to ini files where size constraint issues are not important.*

*Additionally, a **1** usually indicates that an activity is **True** (**Active**) while a **0** usually indicates that an activity is **False** (**Inactive**).*

Note

*Notice that I said UserID logged in as opposed to user. Anyone could log in as anyone else throwing the **Time Clock** off.*

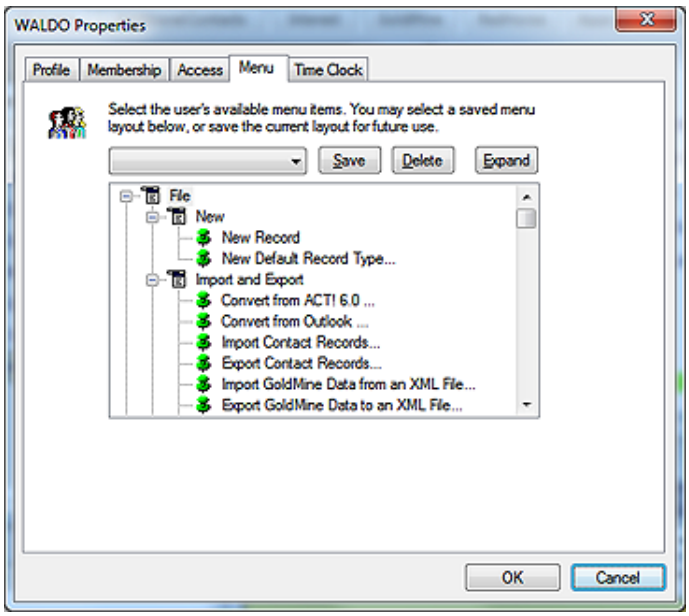


Figure 2-8

screen shot, shown here in Figure 2-8, the individual menu items have a thumbtack icon to the left of each item. The green thumbtack indicates that the menu option is to be displayed on the menu for the user having the correct rights, while a red thumb tack indicates that the menu item is not to be displayed nor enabled. Please note, in Figure 2-8, that I am showing you a screen shot of a menu tree that has been fully expanded. You long time GoldMine users won't recognize this menu any longer as it is a complete revamp of the menu to be more consistent with the Windows definitions for Menus.

As the GoldMine Administrator, you would now begin at the top of the expanded menu tree, and turn off those items that you do not want enabled and functional for this users menu. By default all of the menu items are set to on, and unless you change them, they will have every GoldMine menu item that their user rights level mandates. When done, if this user is a typical user, or a secretary, or in some position where it is possible that the menu created would apply to other users in your organization that you may be configuring, then you would, should, could save the menu for later use. It is recommended that you sit down and define what menu items should and should not be available based upon the users position within your organization. For instances, you may not want the secretary to be able to create a **New Database...**, or to **Maintain Database...**. In general, you probably don't want anyone to be able to **Maintain** the **Database...** except for you or your alternate designated GoldMine Administrator. One would expect that you would do that through the SQL Server Management Studio in an automatic fashion or by hand through the GoldMine menu option.

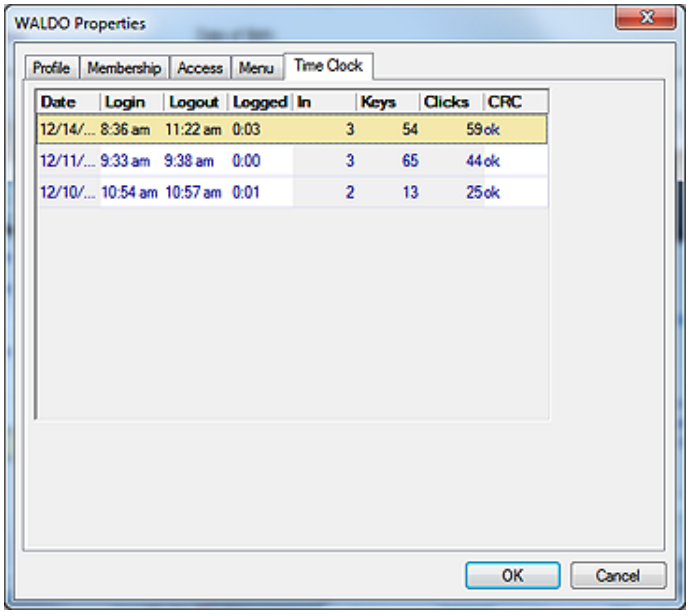


Figure 2-9

I only show you the next tab **Time Clock**, Figure 2-9, because it exists. This is the GoldMine version of George Orwells - **"Big Brother"**. There is nothing that you, as the GoldMine Administrator, can setup under this tab. This will show you when this UserID logged in and out, and for how long they were logged in. It will display the number of keystrokes made during that session, and clicks of the mouse that were performed during the period of time for which the user was logged in. And, lastly, there is the old **CRC** column which will probably just have an **ok** in it. This shows that no one has tampered with these records in the database to make it look as though they were working when, in fact, they were in the parlor with Colonel Mustard plotting the demise of Professor Plum. The information contained in this tab is not reliable. There are times that GoldMine believes that a user is logged into GoldMine, when in fact, they have gone home, and had shut their computer down while GoldMine was still running. Also, it could have been that their computer crashed while GoldMine was running. I wouldn't recommend docking anyones pay based on this information.

User Groups

User Groups are extremely important. As we saw earlier in this chapter, **User Groups** can be used when assigning **Access to Others** activities, or in record **Ownership**. A well defined user group structure will assist the GoldMine Administrator when defining the rights of individual users throughout GoldMine. Also, when established properly, user groups should eliminate the need to give anyone Master Rights, as only the GoldMine Administrators should have Master Rights anyway. To bring up the dialog form, shown here in Figure 2-10, I selected **Tools** | **User Groups...**, and from this dialog form, the **User Groups Setup**, one can setup their various **User Groups**. Many organizations lay their group structure out on paper first, and then transpose that information to user groups through this dialog form. More often than not, however, the structures end up in a pyramid shaped structure with the user groups having the most access being at the top of the pyramid. While those with the least access appear, more commonly, toward the base of the pyramid.

Figure 2-10

The actual setup of the **User Groups** is a rather simple process. One defines a group name, and then adds existing users to this group. Flip a coin. Heads you define your users first, and then build your user groups. Tails you define your user groups first, and then build your users. If you already have your structure laid out on paper, then most GoldMine Administrators find it easier to name their user groups first, and defining the users membership as they add the users in GoldMine. I have reversed the process in this book only because of the way that I have laid out the book, and I prefer to create my user groups after I have created my UserIDs. Doing it the way the book flows, once the GoldMine Administrator has defined the **User Groups**, they would then need to go back and reestablish the user rights. It doesn't really matter which way you choose as long as it gets you to your end goal upon completion.

I won't discuss the **Edit** and **Delete** buttons as they are rather self explanatory. When you click on the **New** button, however, you bring up the dialog form shown here in Figure 2-11. Try to name the user group something meaningful, and representative of the group while staying within the character constraint of the **Enter the User Group's name:** field.

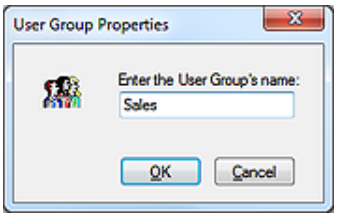


Figure 2-11

I have chosen **Sales** as a user group that I wish to define. Once you have named your user group, you click on the **OK** button to accept the name. The GoldMine application will then add the new user group name to the list of User Groups available for member setup. In fact, once the first user group is defined, the Members Setup button becomes enabled if it was not previously enabled.

Clicking on the **Members Setup** button, as can be found in Figure 2-10, will bring up the dialog form displayed here in Figure 2-12. This dialog form clearly shows two lists on the left hand side, and a single list on the right hand side. The one thing that the dialog form does not display is for which user group you are currently building a group of users. You must remember that piece of information, or manipulate your dialog forms around to see which group is highlighted in the **User Groups Setup** dialog form.

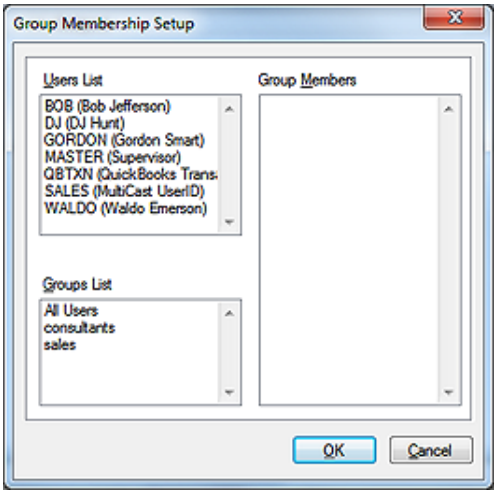


Figure 2-12

You can add **Group Members** to the group in one of two ways. The first is to add members individually one at a time. If that is your preferred method, then you simply need to double-click on the user name from the **Users List** on the left hand side. Sometimes a new group may include all of the members of an existing user group, and maybe one or two others. If this is the case, then there is no need to add the users individually. You may add the users by double-clicking on the user group name from the **Groups List**

Note

Plan for the future.

Even if a user would be a single individual in a group, create the single UserID in that User Group.

*You can always add members later if necessary, and **Ownership**, when being utilized, should be assigned to a **User Group**, and usually not to individuals.*

on the left hand side of the dialog form. This will add all of the members of that particular user group to the **Group Members** list for this group. Fear not, if a user belongs to more than one user group their name will only be appended to this group member list once. You may then add any additional members to the user group from the **Users List**. The observant reader will notice that GoldMine has supplied a default user group of **All Users**. Sometimes it is easier to add all users, and then to remove those UserIDs that should not be in the **Group Members** list by double-clicking on the UserID from within the **Group Members** list.

As in all cases, once satisfied that you have defined the group membership properly, click on the **OK** button to accept the newly created/modified user group membership list. Once accepted, the new members will show up in the **Group Members** list for the highlighted **User Groups**, refer back to Figure 2-10.

If you are following along with this book, and you have finished defining all of your **User Groups**, you would now want to return to the **Users' Master File** dialog form, Figure 2-1, and specifically to the UserID **Properties** (see Figure 2-3), and proceed with the setting of any **Ownership** by group that may be required. Additionally, you may need to re-address the **Access to Others** rights as they may pertain to **User Groups**. This would be under the **Access** tab for the given UserID (refer back to Figure 2-6).

Resources

Recommendation

*By default, any user can add **New**, **Edit**, or **Delete** a resource from the **Resources Master File** dialog form. I recommend that you remove this option from the GoldMine menu for all users except those users having Master Rights (refer to Figure 2-8) or an individual UserID that is responsible for scheduling your corporate resources.*

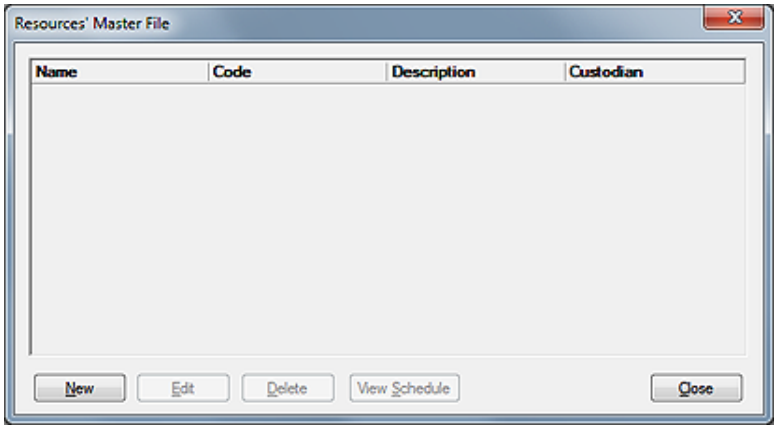


Figure 2-13

Next I venture into the **Resources' Master File** dialog form within the GoldMine environment. The user may bring up the dialog form, shown in Figure 2-13, by clicking on **Tools** | **Configure** ► | **Company Resources...** from the main GoldMine menu.

Note

*You mave remove **Company Resources..** from any UserID menu. This screen, however, will be the only location from which the user may view all of the resouces scheduled, **View Schedule**.*

There are a couple of things that I would like to mention concerning Resources within the GoldMine environment. The first being that, by default, **Company Resources...** appears on every users GoldMine menu regardless of whether they have Master Rights or not. This means that any user can utilize the **New**, **Edit**, or **Delete** option of a resource from the **Resources' Master File** dialog form. I feel that this is a task that is best left to the GoldMine Administrator or to a UserID that has been specifically delegated to monitor corporate resources.

The second item, I would like to mention, is the improved usage of resources within the GoldMine Premium environment. So much so, in my previous books, I actually recommended not using the **Resources' Master File** at all, and, instead, recommended adding the resources as special users within the GoldMine environment. In previous books I recommended adding these as **^Comp1** type of users which forced the resource to the bottom of the **F2 Lookup** list of Users. I made these recommendations as GoldMine would not perform conflict checking against any resources that were scheduled along with the activity. In todays builds of the GoldMine Premium product there is conflict checking, and resolution for users and resources alike. I commend FrontRange on their consistent implementation of user requested changes and features. This is one of the many features that was often requested by the end user, and, consequently, implemented by the coders at FrontRange.

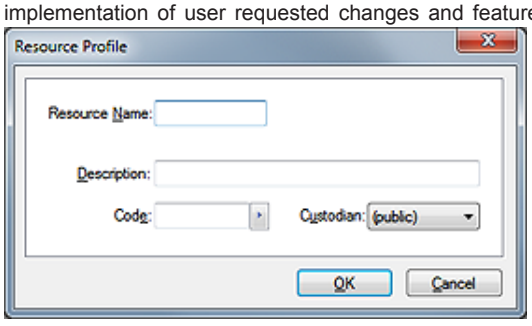


Figure 2-14

Having stated that, and in that I can now endorse the usage of the **Resources' Master File** dialog form for maintaining corporate resources (assets), although, GoldMine is, by no means, an Asset Management System. I shouldn't need to discuss the **Edit** or **Delete** buttons on this dialog form, however, when the user clicks on

the **New** button the dialog form, Figure 2-14 on the previous page, is displayed. In this dialog form, the user may add/edit a new/existing resource. The **Resource Name:** field will accept up to an eight character name for the resource. The **Description:** field will accept up to a forty character colloquial name for the resource. The **Code:** field, while it will accept up to ten characters, is not really designated for any purpose. The GoldMine Administrator may determine that they wish to store the serial number for the resource in this field or even the asset tag number. Another use for this field might be the physical location of the resource when it is in-house. The GoldMine Administrator is free to employ this field in any manner in which they see that fits their organizational requirements. The **Custodian:** field is a drop list of users, and is used to indicate which user, if any, is the custodian of this resource/asset. This article may not be a resource of a user group unless that user group is the **(public)** user group. In the default configuration, GoldMine will default this field to the (public) user (no custodian). The custodian must be the (public) user group or assigned to a UserID. When you are satisfied with the additions/edits to this resource, you may click on the **OK** button to accept the addition/edits for this resource.

The GoldMine Administrator may add as many resources as they deem necessary, and that need to be scheduled along with their users' GoldMine activities. I would point out that resources can not be viewed along with users in the **Planner** tab under the graphical Calendar, however, GoldMine will check for conflicts when scheduling, and will be cognizant of resources when checking **Free/Busy** for an activity that is being scheduled as shown below in Figure 2-15.

Note

*Just to be perfectly clear, one can only schedule a resource when scheduling an activity for a user, and one can only view the schedules of resources from the **Resource(s) Activity List**.*

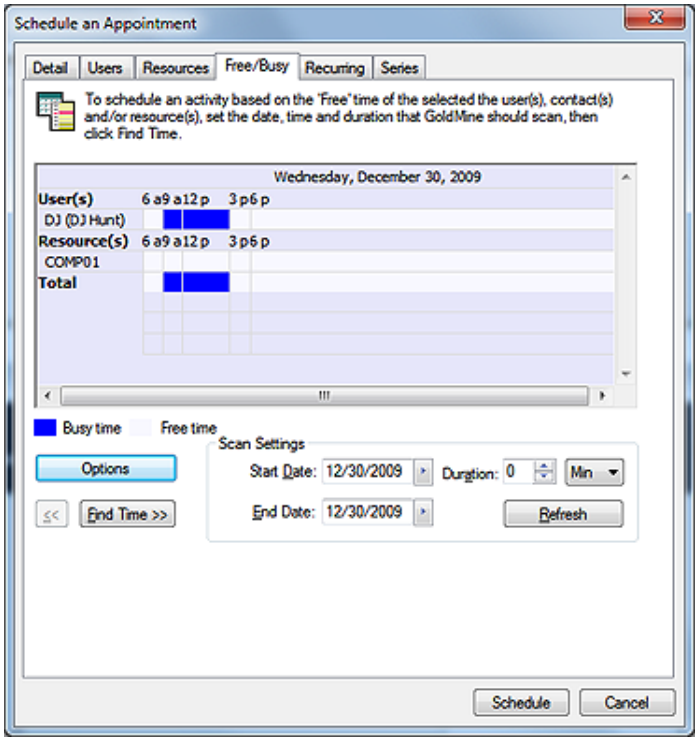


Figure 2-15

On the other hand, if you wanted to view the scheduled resources, you would need to do that through the **Resource(s) Activity List** dialog form as shown below, Figure 2-16.

Note

*The observant will notice that there is no **Contact** name display in Figure 2-16 even though one has been linked.*

I have reported this annomoly to FrontRange, and the issue may be resolved in your build of GoldMine Premium v9.

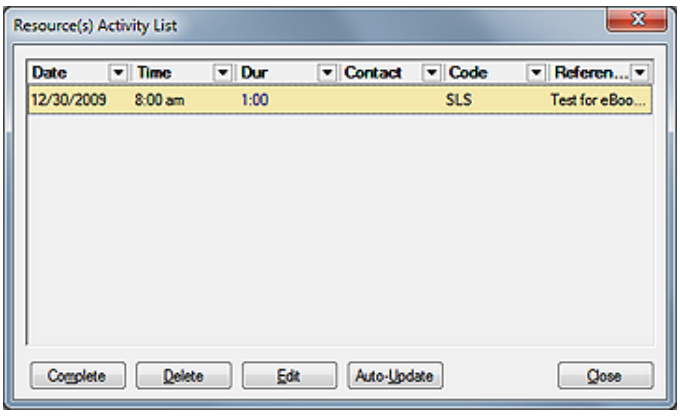


Figure 2-16

From here you may actually **Complete**, **Delete**, **Edit**, or **Auto-Update** the resources scheduled information. Therefore, if you have company assets that do need control, the **Resources' Master File** may assist you in this effort, and I can now endorse the use of this tool within GoldMine.

License Manager

By clicking on **Tools** | **Configure** ► | **License Manager**... from the menu, the user will bring up the **GoldMine License Manager** dialog form, Figure 2-17. By default the user will not even be able to get to this point unless there is at least one license entered to start GoldMine, the Master License. License numbers are broken down into different segments.

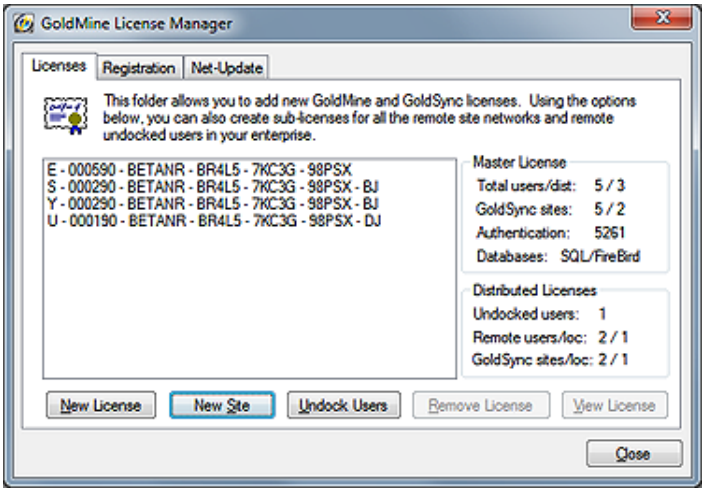


Figure 2-17

Here are a couple of examples of different licenses that I'll talk about later:

E - 000590 - BETANR - BR4L5 - 7KC3G - 98PSX - 6ZY40 - Q3S66 - GDY0G
U - 000190 - BETANR - BR4L5 - 7KC3G - 98PSX - TFYDB - ZYYNU - QXQI2 - 8591 - DJ

Note

*As of GoldMine Premium v9, the **O** - GISMO license has been removed from the GoldMine license coding system.*

The first segment of any license identifies the type of license that it is using, and is a single character:

B - Bump license
E - Premium Edition (SQL/FireBird) license
J - Premium Edition (SQL/FireBird) Bump license
S - GoldMine Site license
U - Undocked User license
Y - GoldSync Site license

The next segment of any license identifies the number of licenses that are included within the given number as well as the version number against which the license is empowered. Using this license number as an example:

Note

*The 3rd segment, in this case **BETANR**, will consist of your HDA as registered with FrontRange, and will need to be utilized whenever communicating with FrontRange.*

Position: **123456**
|||||||
E - **000590** - BETANR - BR4L5 - 7KC3G - 98PSX - 6ZY40 - Q3S66 - GDY0G

You will notice that the first 4 characters of this, the second section of the license number, represents the total number of licenses, in this case **5**, in this the Master License. Characters 5 & 6 of this represents the version of GoldMine against which this license can be utilized.

Note

*You will never see the **KeyCode** in the **GoldMine License Manager** dialog form. You must have the original certificates for the Master license, and you can regenerate the Undocked or Site License to realize the KeyCodes for those types of licenses.*

The next segment, segment 3, contains the owners HDA number as registered at FrontRange. Segments 4, 5, and 6 are the serial number. These 4 segments will be consistent throughout all licenses cut from the Master License, while the last 3 segments of the license are known as the **KeyCode**, and they will not be consistent. Some licenses (Undocked & Site) have two more segments, a 4 character long **Authentication Code**, and a final segment that contains the UserID/Site (all caps) that the license was cut against. Look at the Undocked (**U**) license above as an example of this type of license.

The default license is the Master License while almost all others are derivatives of that license. For example, if I wanted to increase the Primium Edition Master license, previously shown, to thirty licenses, I would need another 25 user Premium Edition Bump license. To do this, I would click on the **New License** button, click on the **I Agree** button to the upgrade statement, and then enter the new license which is either an **E** license or a **J** license.

The refreshed **GoldMine License Manager** dialog form might contain this information after having done so:

E - 003080 - BETANR - 0KUMA - 5MA4B - SMMSG
B - 002580 - BETANR - 46GAV - 9MDQE - KUI9D

The number of licenses in the Master License was bumped up from 5 to 30 licenses, while the newly added license type code was designated as a **B**ump (**B**) license to the Master license. Nothing will decrease the number of Master licenses on the screen, however, some of these licenses may be distributed to other Sites or Undocked users. Looking at Figure 2-17 on the previous page, the reader will see a non-editible frame to the right entitled **Master License**. Here the user can see how the Master License is currently allocated.

For example, as can be seen in Figure 2-17:

Total users/dist:	**5 / 3**
GoldSync sites:	**5 / 2**
Authentication:	**5261**
Databases:	**SQL/FiredBird**

Note

Site Sub-License *does not necessarily imply off-site. You may have a department that wants to have their own GoldMine, and you may wish to cut some of your licenses away for them to utilize.*

By doing this, you will be allowed to synchronize their data into the Corporate GoldMine at any time in the future, and is probably the best alternative to sharing the same database.

Note

Unlike past editions of GoldMine, the ***Site Name:*** *is now limited to eight characters. Hence, my abbreviation of* ***Fitchburg*** *to* ***Fitchbrg*** *in GoldMine Premium.*

Let's take a look at another example. This time, I want to cut a Site license for our location here in Fitchburg, Massachusetts (or another Department within our office). I click on the **New Site** button to bring up the **Create a Site Sub-License** dialog form shown here in Figure 2-18.

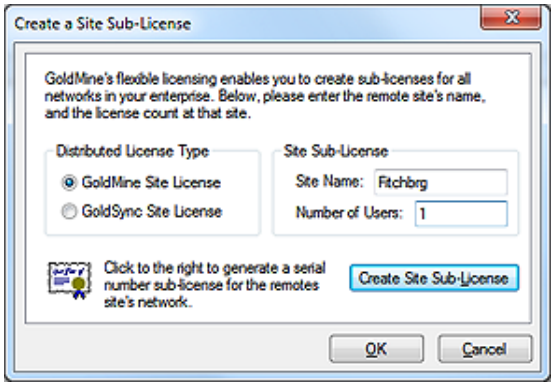


Figure 2-18

I then select the type of license that I wish to distribute, which, by default, is the ◉ **GoldMine Site License**. I use the **Site Name:** field to enter **Fitchbrg**, and the **Number of Users:** to identify how many licenses I wish to distribute to the Fitchbrg site. In this example, I am distributing **1** license. When completed, I click on the **Create Site Sub-License** button to generate a license for that site. The license might look like the one shown here after you click on the **I Agree** button that will present itself:

S - 000180 - BETANR - 0KUMA - 5MA4B - SMMSG - TFYDB - ZYYNU - QXQI2 - 8591 - Fitchbrg

Additionally, the Master License information will have changed. Cutting a Site License will reduce the number of licenses that you have available for your network installation of GoldMine. For example:

Total users/dist:	**5 / 4**
GoldSync sites:	**5**
Authentication:	**5261**
Databases:	**SQL/FiredBird**

That is slightly different than if you were to create an **Undocked User**. This one is a bit harder to explain, but I'll give it a try. If you cut an Undocked User license, you do reduce the number of licenses, in your pool of licenses, by one, however, when that undocked user is on site, and using the network GoldMine, they are not using a license from the pool of remaining licenses. Instead, they are using their undocked license even while on the main system, therefore, all of the pool licenses are still available for your network users of GoldMine.

The **GoldMine License Manager** dialog form, for what I have described so far, might look like the one shown in Figure 2-17. Pay particular attention to the **Master License** information frame, and the **Distributed Licenses** information frame whenever you are adding or removing licenses.

The next tab is the **Registration** tab, and it shows your GoldMine registration information as it should have been submitted to FrontRange upon registration, see Figure 2-19 on the next page. This information is employed when using **MapQuest** through GoldMine to look up directions to a contact record location from the registered location. This information can be changed if it is not accurate, however, there is a little trick to allow you to change this information. You must click on the **Net-Update** tab, and select the checkbox that is shown on that page. First select this option ☐ **Update registration information**, then go back to the **Registration** tab. You will find that the fields under the **Registration** tab are now enabled, and are ready for editing. Once you have made the appropriate corrections they will be maintained. Well that seems obvious. So why do I mention it at all? In past versions of GoldMine this information would only be retained though the active GoldMine session unless one se-

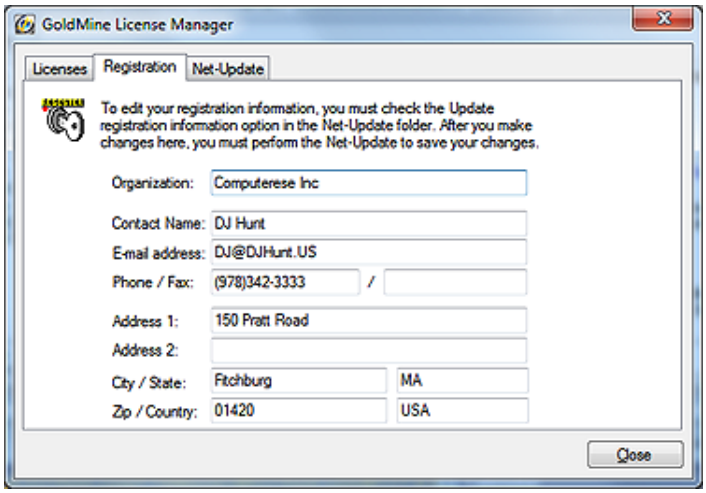


Figure 2-19

lected the **Net-Update Now** button, therefore, this is a change that needs to be discussed.

Previous readers of my Hacker's Guide series of books will remember that there Used to be an **Install Locally** tab following the **Net-Update** tab. It should be obvious from Figure 2-19 that this tab no longer exists. In fact, FrontRange now mandates that you do Workstation installs (Local Installs) from the CD or the batch process if you have configured that properly on the Server from the saved files of the first Workstation install. You should be able to find this information in the last edition of the **GoldMine Install Guide**, a PDF, distributed by FrontRange.

3

In This Chapter

Here is yet another instance where FrontRange has made changes to the GoldMine as we knew it. No longer are these called User Preferences, instead, we now call them User Options. These Options can now be accessed via **Tools** | **Options** selection from the GoldMine menu or from the more readily accessible Standard Toolbar **Options** button.

When an individual is added to GoldMine as a user, the GoldMine Administrator assigns that user certain rights, and capabilities while using the GoldMine product. Beyond that, however, the user may control their own GoldMine environment. These environmental settings are stored in an ini file that resides in the main GoldMine folder, and is preceded by the GoldMine login UserID name created by their GoldMine Administrator. In Example: a GoldMine login name may be DJ, therefore, the related initialization file would be **DJ.ini**, and would reside in the main GoldMine folder. Be careful not to confuse this with **DJ.tbi** which contains the **Taskbar** settings for this UserID. The actual ini file with basic content will not exist in the main GoldMine folder until after said user has logged into GoldMine for the first time. Once the ini is created, it will be modified each time the user exits GoldMine, or, as we will discuss here, as the user makes changes to their **Options**. It is important, therefore, that the GoldMine Administrator does not set the properties of this file to **Read Only**, as we have seen done in some organizations, unless the GoldMine Administrator deems it necessary to not allow the users to make permanent changes to their UserID.ini files. There is no way to prevent the user from making temporary Option changes which are maintained in memory during any active GoldMine session.

I would like to note that the UserID.ini no longer contains all of the default settings as well as user selected options. In fact, it only posses the bare minimum of information upon initialization, and some of that, the **[Background]** section, does not even function with the GoldMine Premium Tabbed or Windowed environment.

```
[Exclusions]
Classic=0
Hash=527
Full=133_140_141_146_154_156_164_297_320_722_

[GM_SEARCH_CENTER]
SyncContact=0
DefField=0
SyncDelay=2
LookByShrink=0
SelectAction=0
FindUSAPhone=1
GM_SEARCH_CENTER=1

[Background]
LogoPosition=RB
LogoMargin=8,10
TextPosition=RB
Background=C:\Program Files\GoldMine\backtile.bmp
BackgroundPosition=TILE
LogoMask=47,79,136
Text=<gm_welcome>
TextFont=Arial
TextHeight=32
TextBold=1
TextColor=234,186,21
init_user=1
TextItalic=1
TextMargin=46,24

[ContactListColInfo]
ColumnHeaderState=ver2;40;COMPANY;1;-1;0;0;40;CONTACT;2;-
1;0;0;25;PHONE1;3;-1;0;0;50;[];4;-1;0;0;20;STATE;5;-1;0;0;20;ZIP;6;-1;0;0;
FilterEnabled=0
SummaryEnabled=0
GroupEnabled=0
```

```
[Warning]
ShowTipOfTheDay=1
```

You will also find that the UserID.ini body, itself, has changed as it now incorporates both straight initialization strings as shown above, as well as a single, albeit humongous, XML string which I hope to discuss briefly later in this chapter.

Personal

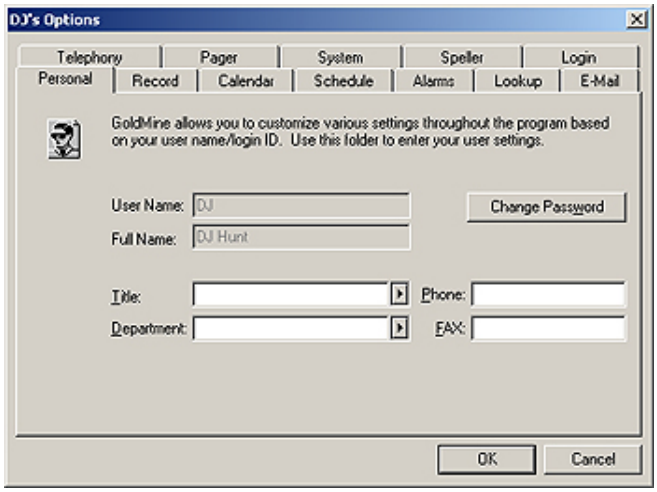


Figure 3-1

Tools | **Options** selected from the main GoldMine menu will produce the GUI (Graphical User Interface) portal to the UserID.ini settings. The first, or **Personal**, tab of this dialog form is shown in Figure 3-1.

One should immediately notice that the **User Name:** field, and the **Full Name:** field are disabled. The GoldMine Administrator added these to GoldMine Premium, and the user does not possess the ability to make changes to either of these two fields. In fact, once added, even the GoldMine Administrator is only permitted to change the Full Name: value, and not the User Name: value. Should the user desire that their full name be changed, as the users full name does appear as displayed here in their document templates, e-mail templates and reports, the user would need to request that their GoldMine Administrator or a UserID login with Master Rights make this change for them. If the user wishes to have their UserID as well as their full name changed, then the GoldMine Administrator would be required to Clone the existing UserID in order to make the changes. The GoldMine Administrator would then be required to do a Territory Realignment before deleting the old UserID.

The remaining four fields add information about the user that may also be used in document templates, e-mail templates as well as reports. This information in stored in the UserID.ini under the section heading of **[User_Var]**. We'll discuss more on this shortly, and there are GoldMine macros specifically created to utilize this data (see Appendix B). It is here, through this GUI, that the user should add their **Title:**, **Department:**, **Phone:**, and **Fax:**. Once this has been accomplished, these user variables would, hence forth, be available for use in document templates, and e-mail templates, as well as in reports. The UserID.ini, at least the section populated by this dialog form tab, might look like the section shown here:

```
[User_Var]
Title = Owner
Dept = Support
Phone = (978)342-3333
FAX =
```

To employ any of these variables in an e-mail template, or a document template, **Title** for example, it is simply a matter of adding the GoldMine macro which is something like **<<&User_Var.Title>>** at the appropriate position in the template.

Although this GUI will only allow the user to add the four user variables mentioned, one may open the specific UserID.ini using Microsoft NotePad and add, by hand, as many other user variables as they would have a need to use in a document template, an e-mail template or in reports. There is no documented limit to the number of user defined variables that one may add by hand, although there is a functional limit to the size of the UserID.ini file itself. Here is an example of what that section might look like for someone else.

```
[User_Var]
Author = Author: GoldMine – A Technical Guide
Author2 = Author:  Sales Force Automation with GoldMine 4.0
Certification = GoldMine® Certified Professional - GoldMine 5.0
Certification2 = GoldMine® Partner
EMail = DJ@DJHunt.US
FAX =
Phone = (978)342-3333
Title = Owner
Dept = Application Development
URL = www.DJ-Hunt.com
```

Note

*Please make note that the **[User_Var]** information is not self formatting, and will appear in your templates exactly as it is entered here, hence, it is incumbent upon your users that they enter the information in a formatted manner consistent with your corporate policies on the matter.*

Tip

*Even though the **[User_Var]** section via the GUI only permits 4 variables, using Notepad, the GoldMine Administrator or GoldMine User with permission may add as many more variables as they need like:*

Email=DJ@DJHunt.US

These variables, like the others, will then be accessible as mergible values in all Document & E-mail templates.

If you have Corporate variables that would be generic to everyones templates, then you may want to consider doing a Corporate Override via the GM.ini as discussed later in this chapter. In Example:

```
[User-OverRide:User_Var]
Company = Computerese
Address1 = 150 Pratt Road
CSZ = Fitchburg, MA 01420
```

Note

*Even though this dialog form is presented to the end user via the Options GUI, this value is **not** stored in the UserID.ini. This value is encrypted, and written to the Users table.*

Reverting back to Figure 3-1 on the previous page, you will notice that there is a button to **Change Password**. By clicking on this button, the user is presented with the dialog form shown here in Figure 3-2.

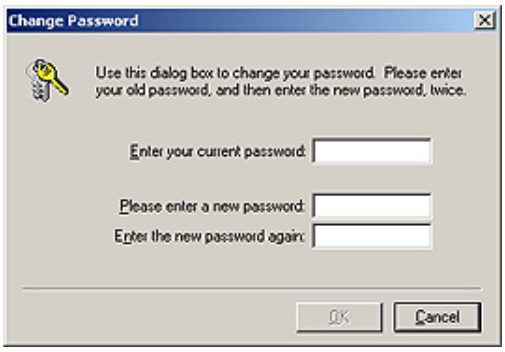


Figure 3-2

This affords the user the opportunity to change their password at any time that they feel it would be appropriate. This is in contrast to the setting that the GoldMine Administrator controls, while creating a new GoldMine user. That is where the GoldMine Administrator would set the users **Valid for days:** password value. This option allows the user to change their password as frequently as they desire.

Note

*Contrary to common belief, the GoldMine UserID Password **is not** case sensitive. For that matter, neither is the UserID Login.*

Once the user types anything into the **Enter your current password:** field, the **OK** button will become active. Once the user has entered their current password, they are then requested to **Please enter a new password:**, and then to **Enter the new password again:**. This is Standard Operating Procedure (SOP) to assure that the user has entered their password as they expected it to be entered, a cross check so to speak. Wisely, the coders at FrontRange have disabled the Copy & Paste functionality of these two fields. Once finished, the user clicks on the **OK** button to have the new password saved for future logins. This will occur if, and only if, the user has entered their current password correctly. If the user does not have a current password, simply leave that field blank. As well, if the user wishes to have no password, leave the two new password fields blank.

Record

Staying on the same row of tabs, and moving to the right, one tab, we now find ourselves on the **Record** tab, Figure 3-3. It is in this tab that the user may set their options as to how they wish to have GoldMine displaying the contact record. On this tab there are three frames of options. There are the **Appearance**, the **ZIP/Postal Code Validation**, and the **Contact Window Title** frames.

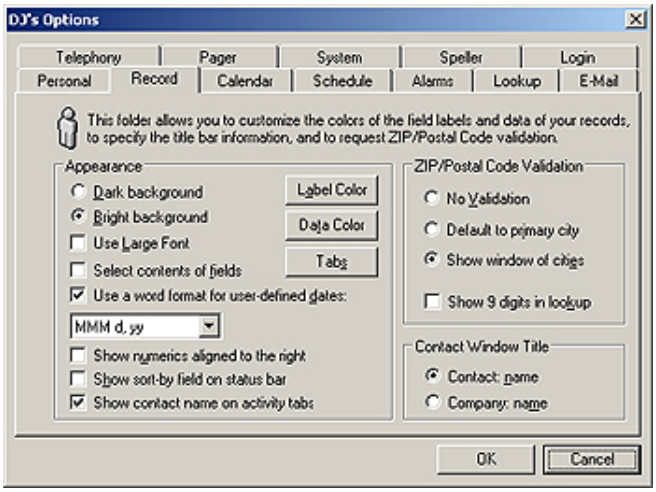


Figure 3-3

I'll look at the **Appearance** frame first, and in this frame, the first option grouping is a radio button selection option. As GoldMine is not consistent in their usage of the radio button selection option, I will explain when GoldMine does not follow the Windows standard for the usage of the radio button selection option. In this case, GoldMine does follow that standard, hence, only one radio button may be selected. The default is ◉ **Bright background**, which uses the Display Properties color of the Window, as defined through the users Microsoft Windows Control Panel, as the background color for the contact record. In most cases this will be the color white. Should the user decide to move away from the default, and to select ○ **Dark background**, then their background will appear in the same color as is showing on the edges of their options dialog form. This color may be gray in an older version of the Windows operating system, or tan in the Windows XP operating environment. In my Windows Vista Ultimate this dark color is a very light grey. This setting is stored in this section of the UserID.ini file.

```
[GoldMine]
;DarkBkGnd=0
DarkBkGnd=1
```

It is a little deceiving in that there is no entry in this section for the default option, and the above entry is only stored in the UserID.ini file when one chooses to use a dark background. However, I would mention that you may change the **1** to a **0** directly in the UserID.ini using Microsoft NotePad, and the user will again have a bright background (see Remarked out statement above preceded with a semi-colon (**;**).

The next item that a user may toggle is whether or not they wish to ☐ **Use Large Font**. I should make it clear that the user is only toggling between the use of the Microsoft Windows registered Small Font, and the Microsoft Windows registered Large Font. GoldMine, itself, does not allow for the manipulation of the fonts with one exception that will be discussed in a moment. The default set-

ting for this switch is **0** (not checked), which means that GoldMine is using the Microsoft Windows, registered, Small Font. As with the light or dark background, nothing appears in the UserID.ini section for the default option, however, should the user select to use the large font, the section would now look like this:

```
[GoldMine]
;LargeFont=0
LargeFont=1
```

In prior versions of GoldMine the:

```
[GoldMine]
BoldFont=1
```

...initialization statement had functioned as indicated. FrontRange has removed this functionality in the GoldMine Premium release that I am currently utilizing to write this book against.

Tip

Remember to use Microsoft Windows NotePad if you plan on manipulating the UserID.ini by hand.

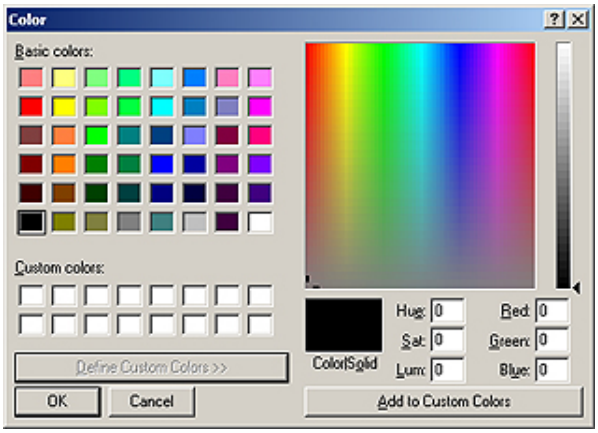


Figure 3-4

Before I continue on with the checkbox option switches, I would like to review the buttons that are available. The first button is the **Label Color** button. Clicking on this button produces the selection dialog shown in Figure 3-4. Obviously Figure 3-4 would be more representative in the printed version of this book if it were printed in color, however, for those of you reading the printed version, if you could just refer to the dialog form on your screen in GoldMine Premium you should be able to understand better what I am about to discuss with you.

Tip

Custom colors are not sticky. If you create 16 different custom colors, they will not be there if you change your mind, and wish to use another custom color. Only define what you want for the one instance.

This selection dialog will let the user choose a default color for all of the labels on their GoldMine Contact record from any of the 48 predefined colors. If the user does not like any of the predefined colors for their labels, they may use the GUI color palette to create up to 16 **Custom colors:** from which to choose a label color.

To select a color, one clicks on the desired color, and selects the **OK** button.

The next button, **Data Color**, will bring up the same dialog, as shown in Figure 3-4, from which the user may choose a default color for all of the data items on their GoldMine Contact record.

Color settings have their very own section in the UserID.ini. This section is not available until after the user has made a change to the default settings. Once a change has been made, however, the section with the appropriate settings will become available in the UserID.ini.

```
[Colors]
DataColor95=8388608
LabelColor95=0
```

Note

Datacolor95 *and* ***LabelColor95*** *are legacy properties that have been incorporated into GoldMine since the product first made its entry into the Windows environment. The property name has never been updated, hence the* ***95*** *attribute assigned to each label for* ***Windows 95****.*

Okay, answer this: How long should anyone honor Legacy products?

Criminey, GoldMine Premium was written to be a Windows Vista product.

The last button, **Tabs**, will produce a dialog form that will allow the user to manipulate the GoldMine tabs. Remember that in GoldMine Premium we have the capability of intermingling the tabs regardless of their type. No longer are the tabs constrained to two separate groups of tabs. Additionally, and new to this edition of GoldMine Premium, no longer can the average user control their own tabs order or names. This button is only enabled for the UserIDs with Master Rights.

In Figure 3-5 on the next page, you will notice that all of the GoldMine tabs are listed (you would need to scroll down the list to see the user-defined tabs). As a comment, I feel that the control over which tabs show, or do not show, and in which order, should reside with the GoldMine Administrator. If a corporate standard is to be set, the GoldMine Administrator must go into each users preferences, and change the tab settings on an individual basis. Today, we can manipulate these tabs for the corporate environment using a User Override which I will discuss later in this chapter under the GM.ini section. The User Override lets the GoldMine Administrator apply a standard for the corporation, and I highly recommend that you develop a standard for your organization. In fact, in this edition of GoldMine Premium, for the Tabs, the User Override can be accomplished directly from within the GUI. I will discuss this in a little more detail in a minute, however, I did want to mention that I think that this is yet another advancement within GoldMine in a positive direction. As you continue on in this book, I think that you see many of these for this edition of GoldMine Premium.

Having said that, and without a User Override, the GoldMine Administrator will have to modify the individual UserIDs Tab settings by modifying the UserID.ini within NotePad. Those UserIDs possessing Master Rights may modify their own Tabs directly, or the Tabs for the corporation via a User Override. Each of the tabs is presented in a list format with the checkbox option box preceding it. A check in any of the boxes means that those tabs will be displayed, while one eliminates the check to remove the tab from their display.

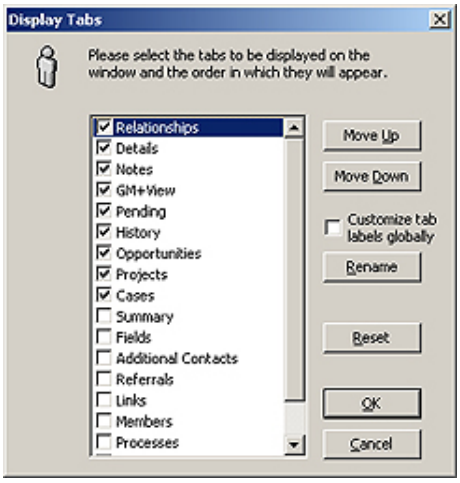


Figure 3-5

To reposition a tab, one clicks anywhere to the right of the tab name that is to be moved, which selects the tab item, and then clicks on the **Move Up** or **Move Down** button, whichever is appropriate. Should one change ones mind, and want to easily restore the defaults, simply click on the **Reset** button at any point during the operation. Doing this will turn all of the default tabs on, and reposition them back to their default position.

Once finished with the arrangement, the user clicks on the **OK** button to write these settings to the UserID.ini. Clicking on the **OK** button creates a new entry in the UserID.ini file as shown on the next page.

```
ROTabItemsOrder=g14;g5;g3;g2;g7;g8;g12;g13;g15;
ROTabItemsHidden=g0;g1;g4;g6;g9;g10;g11;pPC;pPart&ner;
ROTabs1=&Summary,&Fields,GM+&View,&Note,Additional  &Contacts,&Details,&Referrals,&Pend-
ing,&History,&Links,&Members,Pr&ocesses,Opport&unities,Pro&jects,Relati&onships,C&ases,Tic&
kets

[User-Override:GoldMine]
ROTabsGlobalCheck=0
```

Of course, you could always click on the **Cancel** button to abandon all of your changes should you be dissatisfied, and want to begin a new.

New to this edition of GoldMine Premium is the User Override via the GM.ini. You may have noticed the new option checkbox to ☐ **Customize tab labels globally**. With this addition, GoldMine has adapted the GUI to initiate a User Override which, until this edition of GoldMine, has always needed to be accomplished external to the GoldMine application. When a logged in UserID has Master Rights, they will be able to check this box, and modify that tabs corporate wide. This User Override will be discussed later in this chapter in more detail. At this juncture, let's just say that the selection of this option could add something of this nature to the GM.ini as opposed to the UserID.ini:

```
[User-Override:GoldMine]
ROTabs1=&Summary,&Fields,GM+&View,&Memoranda,Additional  &Contacts,&Details,&Referrals,&
Pending,&History,&Links,&Members,Pr&ocesses,Opport&unities,Pro&jects,Relati&onships,C&ases
,Tic&kets
ROTabsGlobalCheck=1
```

With this edition of GoldMine Premium, we also see the addition of a new button which provides you the ability to **Rename** a tab. In this particular case, I had highlighted the Note label name prior to clicking upon the **Rename** button to produce the dialog form shown here in Figure 3-6. Simply enter the new name for this Note tab label (see sidebar Tip), and click upon the **OK** button. You will have changed the name of the Note tab to what ever you had decided upon, and, if the ☐ **Customize tab labels globally** option were selected, you would have made this change corporate wide.

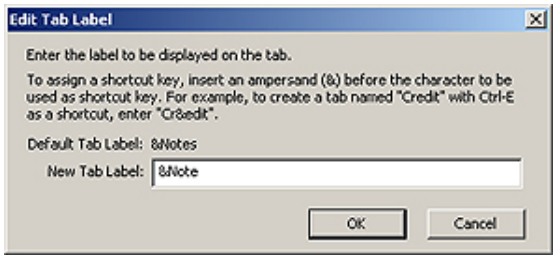


Figure 3-6

Each of the remaining items in the **Appearance** grouping are checkbox option settings, and are either On (if checked), or Off (if unchecked) except for the one drop list box used for selecting a word date display format.

☐ **Select contents of fields**, which, in the default state, is unchecked, allows the user to set the cursor entry type into a field that is activated in the edit mode. By default, when one places a field in the

Note

*An **&** in the tab name indicates the hot-key that a user could keyboard in to quickly advance to the tab, and in GoldMine Premium it is no longer displayed in the Display Tabs dialog form. However, rest assured that the hot-key still works in GoldMine Premium.*

Tip

*As in the past, you may still utilize the **&** hot key marker, however, it is your responsibility to assure that the marker is unique to your system otherwise you may think that you have a hot key that points to this tab when, in point of fact, you have a hot key pointing to somewhere else in GoldMine.*

edit mode, the cursor (I) is positioned at the beginning of the field. If text existed in the field already, and one were to begin typing, that text would be pushed to the right. On the other hand, should the user select this option, and upon entering the edit mode, the cursor would select everything in the field. Any editing to the field, at this point, would cause existing information to be overwritten.

☑ **Use a word format for user-defined dates:**, which, by default, is checked, allows the user to display user-defined date field information in words as opposed to numbers when viewed on the GoldMine screens. Therefore, with the default setting, a user-defined date field would be displayed as MMM d, yy or Nov 23, 48. Should the user change this setting, the date would then be displayed as 11/23/1948. The user may set different word date formats (masks) by choosing a format in the drop list box. The various formats, and the resulting display, are shown here:

MMM d, yy	**Nov 23, 48**
MMMM dd, yyyy	**November 23, 1948**
d MMM yy	**23 Nov 48**
d. MMM yy	**23. Nov 48**
dd MMMM yyyy	**23 November 1948**

☐ **Show numerics aligned to the right**, which, by default, is unchecked, again pertains to how user-defined field information is displayed on the GoldMine screens. This particular setting pertains to the display of user-defined fields having a specific data type of numeric. By default, all data is left aligned on the display. Sometimes, especially when numeric fields are aligned in a column mode, one would desire to have them aligned to the right. Unfortunately, it is all or nothing. Either all user-defined numeric data type fields are right aligned, or all are left aligned. Again, the default, unchecked, is to be left aligned, and it would be up to the user to force the right alignment by checking this option. A User Override would work as well, and force a Corporate Standard.

☐ **Show sort-by field on status bar**, which, also by default, is unchecked, would allow the user to know the order of record sequencing when paging up or paging down through their Contact records. By default, the user must remember the order that was last established. Checking this option will display the order name after the database name in the GoldMine Status Bar to the lower left on the Main GoldMine dialog form display. By default, the status bar might display the database name only **Demo**, however, if this option were selected, the status bar might display **Demo by Company** or **Demo by Contact**. The user needs to determine whether this information is important enough in their daily usage to be displayed, or whether their memory is satisfactory (my memory is not a satisfactory solution).

☑ **Show contact name on activity tabs**, which, by default, is now checked, whereas, in the past the default state was unchecked. The selection of this option does just as stated. When viewing the History tab, Pending tab, or the Activity List, a column will be displayed showing the associated Contact for that activity. This is almost necessary if your organization is using the Additional Contacts on the Contact record (not advised) to display with whom the activity is associated.

Some of these afore mentioned checkbox settings, if changed from the default settings, might appear in the [GoldMine] section of the UserID.ini, although not necessarily in the order displayed, as shown here:

```
ROTitle=1
RoDatesFormat=0
DarkBkGnd=1
RONumsAlignRight=1
ROShowSortBy=1
ShowContactOnActivityTabs=1
ROSetSel=1
LargeFont=1
```

Note

ZIP Code Validation *options set here also affect the secondary contact record functionality as well as the primary contact record.*

Note

I use the terminology of zip code table as if it were a table, in and of itself, when in fact, the zip codes for validation are stored in the ***Lookup*** *table.*

The next frame that I examine is the **ZIP/Postal Code Validation** frame. The first user option in this frame is one of a three radio button selection. The three function as a radio button group option, only one of the three may be selected at any given point in time. Let's examine them now.

○ **No Validation** no longer changes the Contact record tabbing order. Should the user select this option, no zip code validation will take place on entering a new Zip/Postal Code, or editing an existing Zip/Postal Code. Additionally, the user will be able to enter any zip code for any city or state whether it is correct or not. No validation will occur against the zip code table to assure that the zip code is correct for the city and state that has been entered. In my opinion this option is not the recommended option.

○ **Default to primary city** is a bit misleading as no zip code can, within GoldMine, be designated as the primary zip code for a given area. In point of fact, if the user selects this option, then GoldMine will default to the first city that it finds for the given zip code if multiple suburbs are available for the zip code.

◉ **Show window of cities**, is the default selection. Should the user desire to leave this as set (my recommendation) should there be more than one city or suburb for a given zip code, GoldMine would not select the first match from the zip code table for the user. Instead, GoldMine will present the user with a listing of all possible choices from which the user must select the proper city, state combination for the specific Contact record. The user may select an entry from the window by clicking on the item, and then by depressing the **Enter** key, or they may choose to simply double-click the desired entry. Either selection method is recognized by GoldMine.

Lastly, in this frame, is a checkbox option to ☐ **Show 9 digits in lookup**. As you may or may not know, each field in GoldMine has an associated lookup list. The zip code field is no different in this respect. The lookup list itself, however, is presented differently than all of the other lookup lists. The zip code lookup list displays the **ZIP**, the **City**, and the **State** in the lookup listing. GoldMine allows you to add 5 or 9 digit zip codes. The lookup list only displays the first 5 digits of a zip code by default. If your organization embraces the usage of 9 digit zip codes, then your users would be advised to check this box as, when selected, the lookup window will display the full 10 character zip code. i.e. **01420-4142** If not, then you may have multiple suburbs with the same five digit zip code in the listing window, and the user will have no idea as to which may be the correct zip code for the given suburb.

With that, I conclude my discussion of the ZIP Code Validation frame, however, I would like to show how these settings may appear in the UserID.ini under the [GoldMine] section. These are the settings that relate to this grouping:

```
ZipValid=2
Show9DigitZip=1
```

Note

*This particular entry is useful when the user displays multiple contact windows at the same time using the **Window** | **New Contact Window** option from the GoldMine menu.*

Our next, and final frame under the Record tab Options settings is the C**ontact Window Title**, and is a simple radio button selection for which item the user would prefer to display in the Contact record window title bar. Would the user prefer to see the ◉ **Contact: name** or the ○ **Company: name**. The default is to display the contact name in the Contact record title bar (windowed mode)/tab (tabbed mode). Should the user choose to have the company name displayed instead, then in the UserID.ini under the [GoldMine] section you would expect to see:

```
[GoldMine]
ROTitle=2
```

Calendar

Continuing on, we can now explore the user options for the **Calendar** display and functionality which is under the **Calendar** tab. The dialog description is:

"This folder allows you to set your Calendar's display, the scan interval to add/remove updated activities, and which activities are to be automatically Rolled-over."

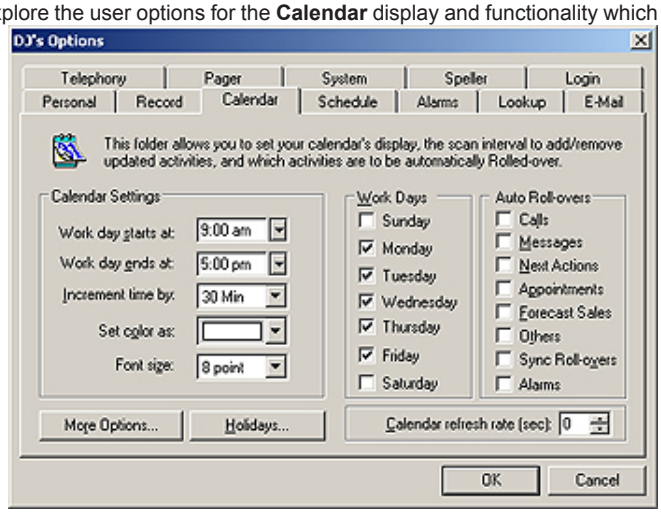


Figure 3-7

Again, I will discuss one frame at a time. The first frame to be discussed is the **Calendar Settings** frame. In this frame, refer here to Figure 3-7, are various display attributes. The first three are separate attributes for one display setting. As simply put as possible, the first and second attribute define the period for having the calendar appear in the color chosen under the fourth attribute. All other times will be displayed as a variant of the selected color. This allows for the visual separation of the work day versus the non-work day on the users graphical Day calendar. The third attribute defines the granularity of the time display on that graphical Day calendar.

The first attribute to set is the **Work day starts at:** time, the time at which the user begins their normal business workday, and the beginning of the color change on the graphical Day calendar. While the second attribute is the **Work day ends at:** time, the time at which the user normally ends their business workday, and the time that the color on the graphical calendar changes back to the designated off work hours color. The next option is to set the calendar display increments using the **Increment time by:** option.

There are various granularities that the graphical calendar may be displayed in:

05 Min
10 Min
15 Min
20 Min
30 Min
60 Min

Next, the user must select the color, from the drop down color list of 10 colors, that they wish to represent on their workday time frame on the graphical calendar. This is done in the Set color as: attribute setting.

The next display attribute determines the Font size: that is to be used when displaying text on the users graphical calendar. The user has but two selections, 8 point or 10 point. Though I make no recommendation, it has been my experience that most users leave the default setting of 8 point for this attribute.

These five attributes could be represented in the UserID.ini in the following section:

```
[CalObj]
DayBegin=09:00
DayEnd=17:00
TimeIncrement=15
SolidColor=65535
FontSize=8
```

Note

*Times in the **[CalObj]** section of the UserID.ini are based on a twenty four hour clock, or what is commonly referred to as military time.*

Referring to Figure 3-7 again, you see that there are no more attributes to be set in this frame, however, there are a couple of buttons that remain to be addressed. The first is **More Options...** and the second button is **Holidays...**. I mention both of these buttons together as clicking either of them will take you to the same dialog form, albeit to a different tab on that dialog form. Let's begin by addressing the **Holidays...** button first.

Clicking on the **Holidays...** button will bring up the dialog form shown below in Figure 3-8.

Clicking on the **New** button, in the **Holiday Categories** frame, will result in the dialog shown in Figure 3-9. The user may also employ the **Import/Export** button in this frame to go from/to an iCal-

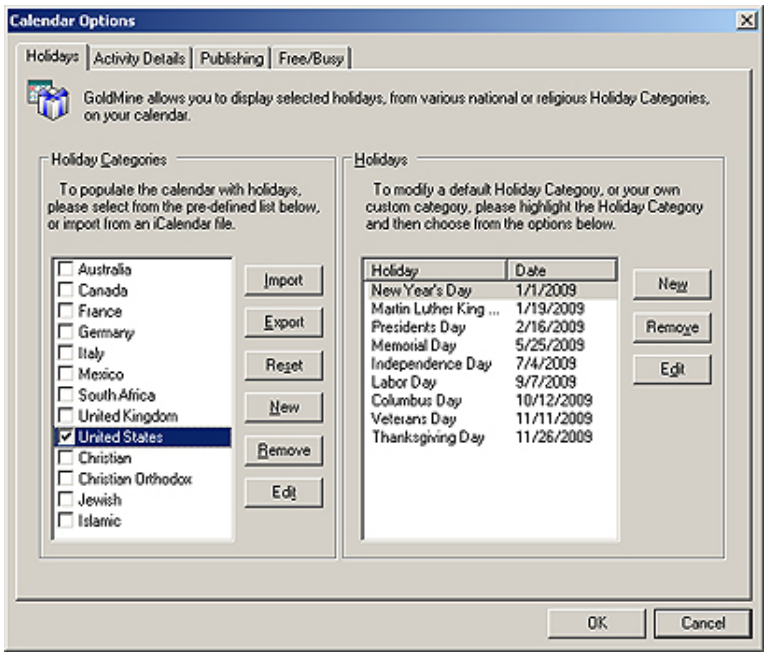


Figure 3-8

endar (*.ics) file. Creating a new holiday category is as simple as typing a name for the category, and then selecting the **OK** button.

Figure 3-9

Once the category has been added, the user may want to add holidays to this category or to any other selected category. This may be accomplished by clicking on the **New** button under the **Holidays** frame, to the right, which will produce the dialog shown in Figure 3-10. I will enter **Personal** into this dialog to create a category for, you guessed it, Personal holidays.

Note

*Only users possessing **Master Rights** will be able to manipulate the **Holiday Categories** or the **Holidays**. All other users will only be allowed to select from the predefined lists.*

Once a category is added to the list, it immediately becomes the selected category for which one may now add holidays. Let's add a holiday to the category Personal by clicking on the **New** button under the **Holidays** frame. Notice that the information in Figure 3-10 has already been added. This holiday is called Patriot's Day, so I have entered that into the Holiday: field. This holiday Repeats every year, and, for this exercise, I have left the setting at **Every 1 year(s)**. It only occurs on one day so I have left the **Duration** at its default setting of **# Days: 1**. This holiday always occurs in the month of April, so I have entered that into the ◉ **On** field along with placing the **19th** in the accompanying field. Please notice that there are two frames here, **By Specific Date**, and **By Day of the Week**. Only one of these frames is available at any given time, and which one is controlled by the selection of the radio button in that frame. I accept this configuration by clicking on the **OK** button.

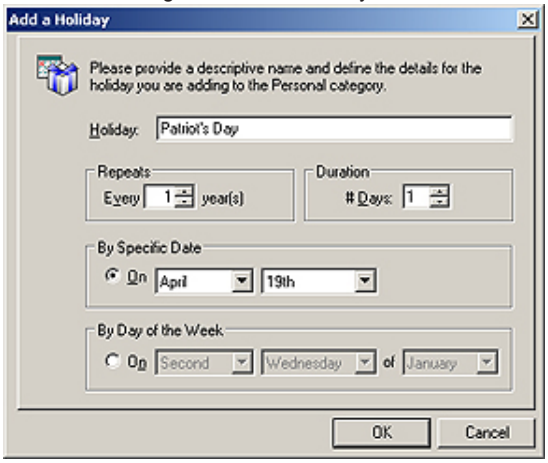


Figure 3-10

If a holiday were two days long, then I simply change the **Duration # Days** to **2**, Figure 3-11. Notice that I used the second frame **By Day of the Week** to establish the first day of this two day, for us, holiday.

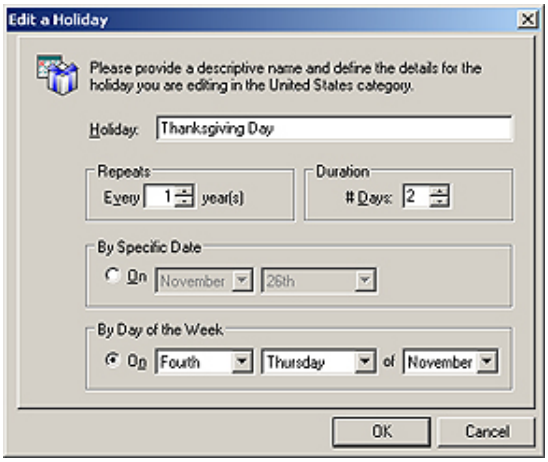


Figure 3-11

So where is all of this holiday information stored? That is the big question. The categories and their contents, as designed by a user possessing Master Rights, are stored as records in the Cal table, having a Cal.RecType of H. The category name, itself, is stored in the field Cal.Ref, while the holidays that are contained within the category, are maintained in the Cal.Notes memo field. This table, and others, are discussed in detail later in this book, in the chapter titled The Tables. Whereas, the holidays that the user, themselves, select are referenced in the UserID.ini in the following section and statement:

[CalObj]
Holidays=8,201

The numbers after the equal sign represent the Cal.UserID number associated with the categories selected by the user. GoldMine uses this indexed field to quickly find the holiday categories selected by the user, and to then load them onto their graphical calendar.

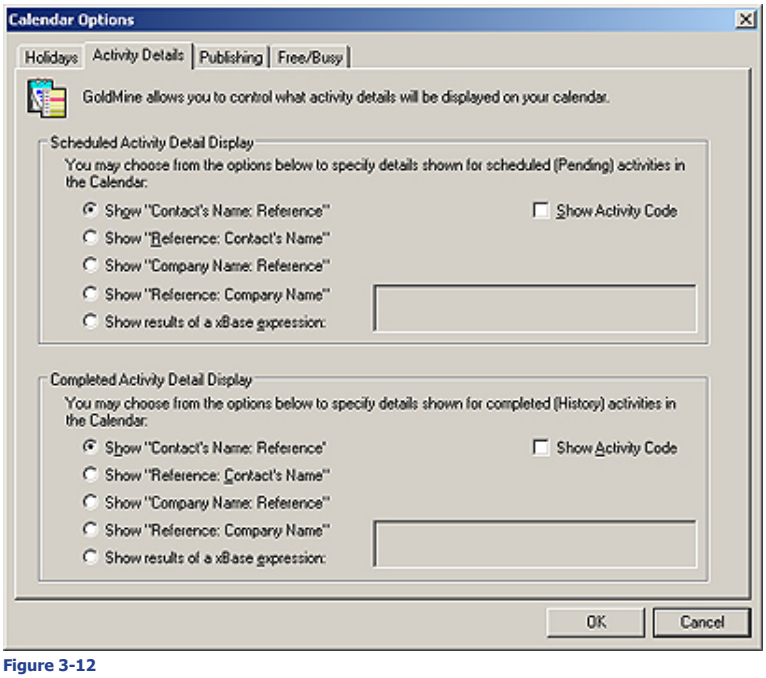


Figure 3-12

The final button that I must discuss from the **Calendar Settings** frame, Fig-

Note

As of this writing, I have not been able to figure out a way to automatically set Monday holidays except one year at a time. i.e. In Massachusetts, Patriot's Day is the 19th of April, but it is celebrated on the Monday preceding the 19th or the 19th if that falls on a Monday.

Note

*The numbers after the equal sign, represent the numbers in the **Cal.UserID** field. For instance **Holidays=8,201** would have the values of **008** and **201** stored in the **Cal.UserID** field of separate records. Old timers will notice that user defined holidays have a new designator. What used to be 011, is now designated 201. Refer to the chapter **The Tables** for more information.*

ure 3-7, is the **More Options...** button. Clicking on said button will produce the dialog form shown in Figure 3-12 as shown on the previous page. One could also have simply clicked upon the **Activity Details** tab as previously shown in Figure 3-8 to arrive at the same location. You are presented with two frames that you may configure from this dialog form, the **Scheduled Activity Detailed Display**, and the **Completed Activity Detail Display**. The former is for designing the display information on the graphical calendar while an activity is Pending, as the latter is for designing the displaying of information on the graphical calendar once an activity has been placed in History. By default, the contact name and the subject of the activity are shown for the Scheduled Activity Detail Display as well as the Completed Activity Detail Display, when they are displayed on the calendar. These text items will appear on the graphical calendar just after the icon which represents the activity that was chosen.

As we have seen in the past in this book, and as we will see again later in these chapters, the radio button selection is a grouping of three buttons. One may accept the default ◉ **Show "Contact's Name: Reference"** or one may select the reverse presentation of that as in ○ **Show "Reference: Contact's Name"**. As I mentioned earlier, these refer to the contact name associated with the activity, and the subject of the activity that has been scheduled. New to GoldMine Premium was the ability to ○ **Show "Company Name: Reference"**, and ○ **Show "Reference: Company Name"**. You may have noticed that there are no hot links for these two newly added radio options. Alternatively, one may choose to be creative, and design your own display, through the use of an xBase expression, by selecting the radio button to ○ **Show results of a xBase expression:**.

Tip

Should one choose to create their own expression for the displaying of calendar information, one is reminded that there is limited viewing space on the calendar, and that there is always the potential of having more than one activity occurring in the same time frame. Don't be overly verbose.

Should the user select to build their own expression, for display on the graphical calendar, they have a wide selection of fields, and characters from which to choose. Let me give you an example of an expression:

```
trim(Contact1->Contact)+[  ] +chr(127)+[ ]+trim(Contact1->Phone1)+[ :o)]
```

...would produce this result in the graphical calendar:

Figure 3-13

I am sure that your users will be more creative. The example demonstrates the use of functions, fields, and text for the desired display results. You may use any valid xBase function as defined in Appendix A of this book. The first thought of most users is that they could use this option to display other information on their graphical calendar. I would like to express an alternative view. This option could also be used to remove the clutter of information from the graphical calendar. This is just another way of looking at the expression option, yet one that I would not want your user to overlook.

Note

*The user determines which activities, Scheduled (Cal table) and Completed (ContHist table), display on their graphical calendar by right-clicking on the **Day** calendar in the timed area, and selecting **Activities...** from the local menu.*

*From the resulting dialog form, **Select Activities to View**, the user checks those activity types that they wish to have displayed, while unchecking those that they do not want displayed on their graphical calendar. My users will uncheck Messages as these do little more than clutter the graphical calendar unnecessarily.*

On the Activity Details tab, Figure 3-12, you also have a simple checkbox option which would allow the user to ☐ **Show Activity Code**. This option displays the activity code should you use Activity Codes (something that I always recommend), from the **Code:** field of the scheduled activity, between the activity icon and any text that is being displayed. Figure 3-13 shows the display when this option has been selected by the user (**[TST]**). Please note that the Activity Code appears after the icon for the activity, but before anything that the user has indicated that they would like to have displayed.

Note

This is one of those areas that the GoldMine Administrator may want to present a corporate image, and to achieve this, the GoldMine Administrator may select to provide a user override as discussed later in this chapter.

At this point I should discuss the **Completed Activity Detail Display** frame, shown in Figure 3-12, and discuss its options. As it is a duplicate of the **Scheduled Activity Detail Display** tab, I do not feel that I need to waste the extra trees that would be required to re-discuss the material. I would like to mention, regardless of what display options the user selects under the Completed Activity Detail Display frame, the activity will be displayed on the graphical calendar with a line through the activity as well as being displayed with a grey background. Additionally, for the Completed Activity Detail Display to be displayed at all in the graphical calendar, the user would have to set options through the calendar itself as to which activities, scheduled and completed, they wish to have displayed (see sidebar Note). This section only lets the user define how an activity, scheduled or completed, is to be displayed, and not which of the activities should be displayed on the graphical calendar.

Both of these settings could be presented in the UserID.ini:

```
[CalObj]
CalShowActvCode=1
CalExpr=trim(Contact1->Contact)+[  ] +chr(127)+[ ]+trim(Contact1->Phone1)+[ :o)]
HistExpr=0
HistShowActvCode=0
```

I will mention here that a **1** in the **CalShowActvCode** means that the **Activity Code** will be displayed when viewing pending activities while the **0** in the **HistShowActvCode** statement means that the **Activity Code** will not be displayed for historical activities. The **CalExpr** and the **HistExpr** state-

ments work slightly different. I had created an expression for our calendar example, therefore, that is included in the **CalExpr** statement. If I had accepted the default radio selection of ◉ **Show "Company Name: Reference"** then this statement would be similar to the **HistExpr** statement. On the other hand, if I had chosen the ○ **Show "Reference: Company Name"** radio button, then the statement would have appeared as **CalExpr = 3** instead. It could get confusing as you can see.

In past versions of GoldMine, this would conclude our discussion in this area, however, one will notice that there are two additional tabs as shown in Figure 3-12. Though these are not necessarily accessed from any buttons on the **Calendar** tab under a user options, they are here and available for discussion. Additionally, any changes made to either section will update the UserID.ini, therefore, I have selected to discuss them here.

Note

When you downloaded this book, a zip file of the web display pages was included in that download.

*You should unzip these pages and put them on your website. I have placed them under **http://www.DJ-Hunt.com/html/DWDSite/iCal** on my site, and, as you will see in my discussion, I will publish to the calendars folder under the **iCal** folder via the **File Transfer Protocol**.*

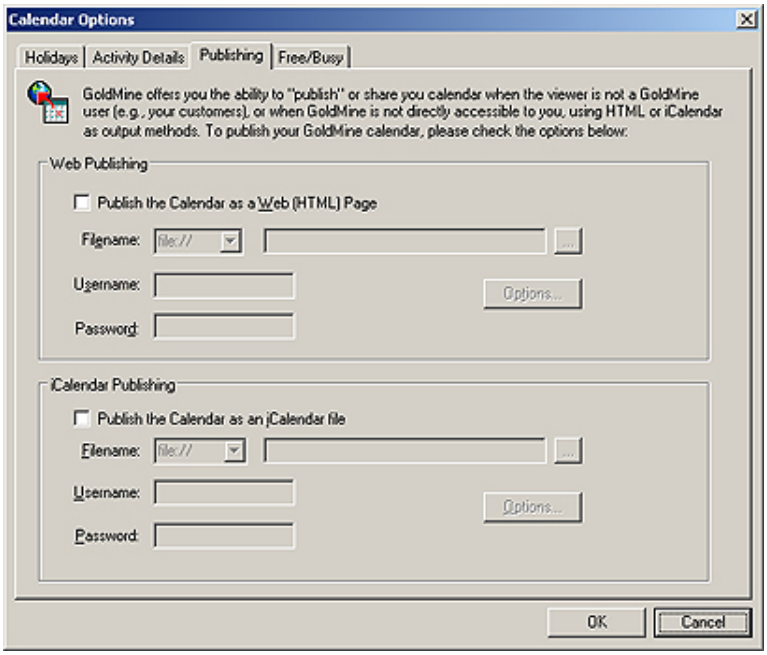


Figure 3-14

The first of these that I will discuss is the **Publishing** tab which I personally use to publish our calendars to a web location. From an earlier version of the GoldMine Help files: *"GoldMine offers you the ability to "publish" or share your calendar when the viewer is not a GoldMine user (e.g., your customers), or when GoldMine is not directly accessible to you, using HTML or iCalendar as output methods."* I have chosen to use the **iCalendar Publishing** frame for this discussion. I have supplied you a zipped download with iCalendar web pages for you to use to display your published iCalendar if you so choose. This is the very same parser that I am currently using on my website. In fact, everything in this section is from my real use of the GoldMine iCal publishing capabilities. I have gotten many requests from GoldMine users asking me how I utilize this very feature. I would add that the steps and the dialogs are identical for **Web Publishing** as they are for **iCalendar Publishing**, and that the only difference is in the published results. **Web Publishing** produces an HTML output calendar, whereas, **iCalendar Publishing** results in an ICS output calendar. As I have stated, my organization has chosen the latter, and it is that which I will now describe.

As you may have noticed in Figure 3-14, you must select the option to ☐ **Publish the Calendar as an iCalendar file**. This enables that frame for further input. In the 1st of the **Filename:** fields, I will choose **ftp://** from the drop list which contains **file://**, **ftp://**, or **http://**, and as my site requires the File Transfer Protocol, I would choose the **ftp://** option. In the 2nd of the **Filename:** fields, I have entered the path, and name on our website to which I want to publish our calendars file, **DJ-Hunt.biz/iCal/Calendars/DJ Hunt Eastern.ics**.

In the **Username:** field I have entered my ftp:// login name, **DJHunt**, as I have entered my password for the ftp:// login into the **Password:** field, ****** (You didn't really think that I would give that out now did you?). I then click on the **Options...** button which brings up the dialog form displayed in Figure 3-15 on the next page. These options are iCalendar filtering conditions. The first tab, **Users**, lets one select the users or user groups for which one desires to publish a calendar. By default only the user for whom the options are being set is selected, however, one may choose any number of users or user groups. Double-clicking on a name or user group will remove it from the list, and place it in the opposing list.

Clicking on the **Activities** tab allows one to select those activities from the **Cal** table that they wish to have published. This filter is slightly different for the HTML versus the ICS publishing format. As

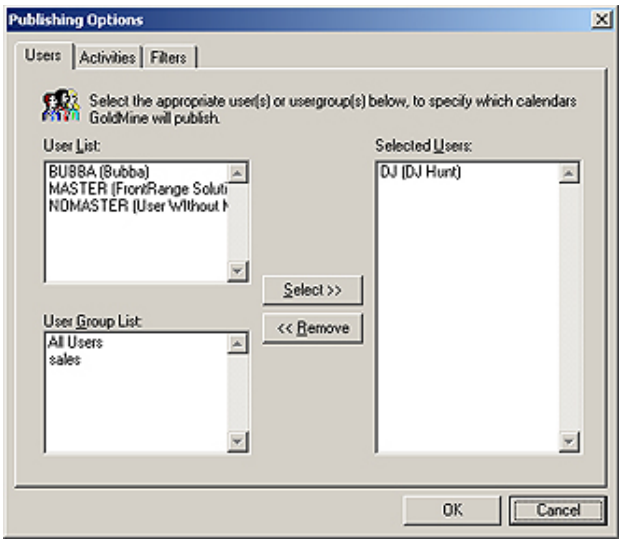


Figure 3-15

you can see in Figure 3-16, there are 5 activities that may be published for ICS. For HTML publishing (not shown) there are 9 calendar activities that one may publish, and 8 historical activities that you may publish as well. The HTML publishing is more akin to the GoldMine graphical calendar than is the ICS publication. I present this to you as it may be a factor in your decision to use HTML over ics publishing. Everything that can be displayed on your graphical calendar can be published to your HTML calendar. As you can see in Figure 3-16, I have only chosen to publish Appointments, Calls, Next Actions and Other actions. I have done so because we never utilize Events by mandate.

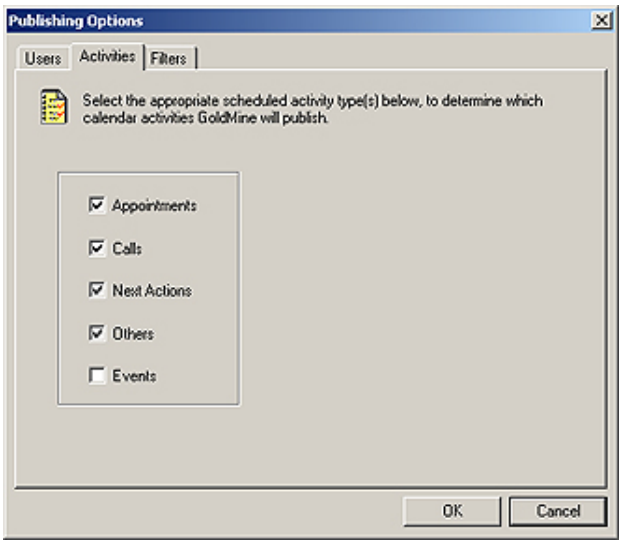


Figure 3-16

All of the afore mentioned options are filters of sorts, however, there is also a **Filters** tab, as shown in Figure 3-17 on the next page, and it is on this tab where you may set to publish a specific date range, or publish the ☐ **Activity Code**, or to publish the ☐ **Reference contains**, or to ☐ **Publish only linked calendar data**. You may choose all, or none, which ever is your preference. As you can see from Figure 3-17, I have chosen to publish within a certain date range, and only those activities on my calendar that are linked to a contact record will be published.

When finished, you simple click on the **OK** button, and then again on the next **OK** button to write the information back to the UserID.ini.

The following information is written to the section shown in the DJ.ini for the options that I have selected:

```
[CalObj]

PublishHTML=0
PublishHTMLPath=
PublishHTMLUsersList=
PublishHTMActvTypes=00000000000000000000000000
Publish2HTMFilterByDate=
PublishHTMFilterByLink=
Publish2HTMStartDate=
Publish2HTMEndDate=
PublishHTMLUser=
PublishHTMLPwd=

PublishICal=1
PublishICalPath=ftp://dj-hunt.biz/ical/calendars/dj hunt booktest.ics
PublishICalUser=djhunt
PublishICalPwd=9EB2C3A13DB38D83B94787AD9EB05759A7D9D5D79D9287AD
PublishICalUsersList=DJ;
PublishICalActvTypes=11100001000000000000000000000
Publish2ICSFilterByDate=30647
Publish2ICSStartDate=20090220
Publish2ICSEndDate=20100220
PublishICSFilterByLink=1
```

We can move on now to the **Free/Busy** tab as shown in Figure 3-18. In this case, *"GoldMine offers you the ability to "publish" your Calendar's Free/Busy time when the viewer/scheduler is not a GoldMine user (e.g. they are an Outlook user), or when GoldMine is not directly accessible to you."*

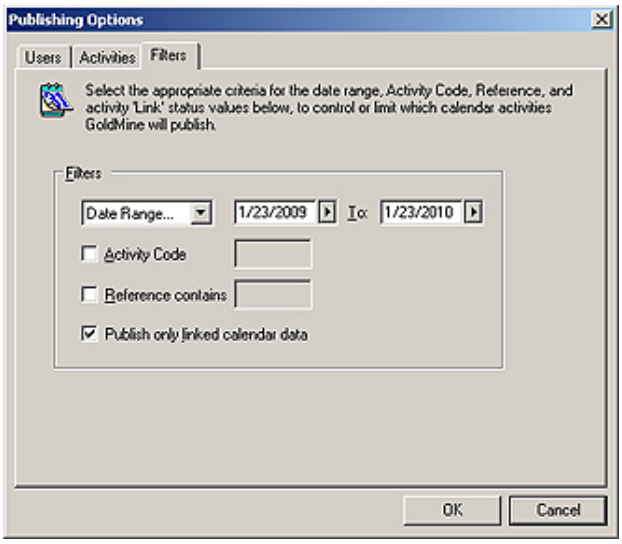


Figure 3-17

Interesting! Basically, this is no different than the Publishing which I just discussed with you. It is an **iCal** type publishing process, as was my previously discussed Calendar Publishing, that could be retrieved into Outlook or any similar iCal compliant application. The Free/Busy tab contains every option under this one dialog form. There is not a separate dialog form, for instance, for Filtering. Everything is inclusive on this one tab.

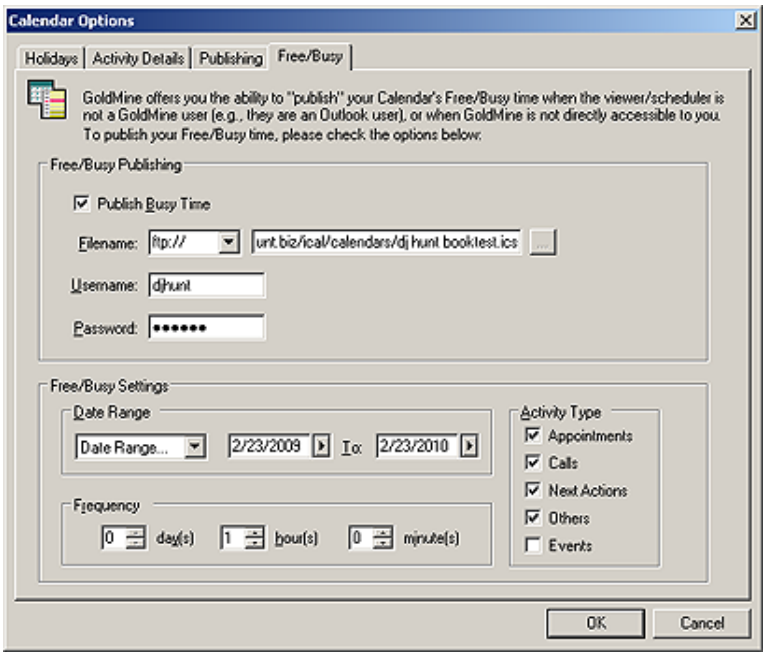


Figure 3-18

In order to enable the fields for data input, one must first select to ☐ **Publish Busy Time**. Once that option is selected, the remainder of the fields on this tab will be enabled for data input. There is very little that is different on this dialog form than was on our previous multi-tab dialog form. The **Filename:** field is first, and it too is a drop list containing your choices of transport for the published material **file://**, **ftp://**, or **http://**. Then there is a following field for the path, and file name, to which you want the published material to be sent. For this exercise I have chosen to include the same information as I previously utilized.

The **Username:** and **Password:** fields will be available if you are publishing to an **ftp://** site, and no further explanation should be necessary for either of these fields, as their labels say it all.

The filter shows up in the **Free/Busy Settings** frame of this dialog form. As before, one may pick a specific period under the Date Range frame, or you may supply a **Date Range...** by picking the start and the **To:** dates from a drop list.

Additionally, as before, one may select the **Activity Types** to be published. From here, we see something that we had not previously seen, **Frequency**. Whereas, pushing **Web Publishing** and **iCalendar Publishing** were on demand, **Free/Busy Publishing** can be somewhat automated. In

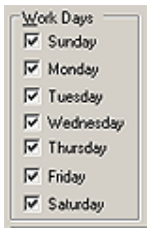
this frame one would set the frequency in **day(s)**, **hour(s)** and/or **minute(s)** that they would like to have GoldMine automatically publish their Free/Busy calendar. This really should be a corporate decision, however, in lieu of that, the user may decide that publishing once a day is sufficient. Possibly, once every 4 hours would be better. I make no recommendations in this instance, as it is truly dependant upon corporate standard operating procedures.

For this exercise the resulting UserID.ini settings might be as shown here:

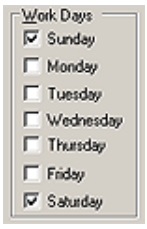
```
[CalObj]
PublishFBPath=ftp://dj-hunt.biz/ical/calendars/dj hunt booktest.ics
PublishFBUser=djhunt
PublishFBPwd=92A8C7E541B79187BD4BA89B92A4EBDDD9DBAB9E9184A89B
PublishFBFilterByDate=30647
PublishFBStartDate=20090223
PublishFBEndDate=20100223
PublishFBFreq=60
PublishFBActvTypes=11100001000000000000000000000
```

This would conclude our examination of the **Calendar Settings** frame, and allows us to proceed to the **Work Days** frame as shown back in Figure 3-7. Simple put, these are checkbox options that work similar to the time settings discussed in a previous section. One checks the days that are considered to be work days, and GoldMine displays the work days, work hours in the color selected in the previous grouping. All of the non work days, and non work hours are displayed in a darker hue of the color selected. By default, the normal work days of Monday, Tuesday, Wednesday, Thursday and Friday are checked. If your user works on days other than these days, they would check and uncheck the options as is appropriate for their need. The results of the selections in this grouping are derived and displayed, in the UserID.ini as shown here:

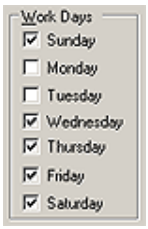
```
[CalObj]
Weekends=65
```



Weekends=0
Figure 3-19a



Weekends=62
Figure 3-19b



Weekends=6
Figure 3-19c

See the sidebar for an explanation as to the derivation of these numbers. Figures 3-19a, 3-19b, and 3-19c show a few more samples of various settings that could have been derived for this section.

We can now delve into the **Auto Roll-over** frame of Figure 3-7. This is a most interesting frame, and sometimes causes consternation in an office. By selecting an activity for auto roll-over, one is stating that should the user not complete an activity on the assigned day, the next time the user logs into GoldMine, said activity will be forwarded to the logged in date. Technically, GoldMine is expecting each activity that is scheduled will also, at some point, be completed creating a history for said activity. I have found, however, users are not as attuned to following procedures as we might want them to be, hence, the need for the corporate establishment of an **Auto Roll-over** paradigm.

Should a user forget to complete an activity, or outright just not complete the activity, then the auto roll-over feature pushes the activity in their face. The users calendar will soon get cluttered on the current date with uncompleted activities and, most likely, the users will complain to you that their calendar is so cluttered. Another complaint that I have seen is: "How come there is all this activity when I first log into GoldMine? It takes forever, to start.". This could also be a result of the Auto Roll-over feature. As one logs into GoldMine, any activity type that is marked for roll-over will be rolled over at the time of the login. Now, when that user checks their calendar, all of the uncompleted activities from prior days will show up on todays calendar.

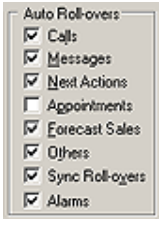


Figure 3-20

Here in Figure 3-20, I am just looking at this one frame so that you won't have to flip back too many pages. Notice the checkbox, ☐ **Sync Roll-over**. If the user, or if you should select for the user to have any activities rolled over, then it would be wise to check this option. By default, as GoldMine is rolling over an activity, it does not update the sync logs with the new date, hence, anyone, out in the field that is synchronizing with the corporate office, will be unaware of any of these scheduled changes. However, let me make a point. Your Remote users should have this option checked for their UserID on their side only, and not for their in house Options. The Remote user should also be encouraged to Schedule and Complete their

Note

Web Publishing *and* ***iCalendar Publishing*** *may be automated as well, however, this must be accomplished via the* ***Server Agents****. One must remember to* ***Start Server Agents****. This may be done via the menu, by hand, or via a keyboard macro when GoldMine is started.*

G:\GoldMine\GMW.exe /u:UserID /p:Password /m:MacroNumber

This is the method that we prefer, however, at least in build 70606 of GoldMine Premium, the keyboard macros are still not functioning under a Vista environment.

Note

The ***Weekends*** *variable setting is based upon the sum of the corresponding values of the* ***unselected*** *work days:*

Sunday	=	***1***
Monday	=	***2***
Tuesday	=	***4***
Wednesday	=	***8***
Thursday	=	***16***
Friday	=	***32***
Saturday	=	***64***

In our case, Figure 3-7, Sunday and Saturday were not selected (default state) as work days, and the resulting statement was:

Weekends = 65

Tip

GoldMine was designed around the paradigm that a person that can schedule their day will be more productive than one that does not. Along with the scheduling of activities, and even if you don't schedule activities, the completion of those activities, and even unscheduled activities, is what creates the history with your customers.

It is important that you get your users in the habit of scheduling everything which they can, and completing everything, scheduled or not.

activities only from the location from which Sync Roll-over has been selected. It is possible that the Remote user could schedule something for this user in a used time slot thinking, and rightly so, that the time was available. If your users activities are being rolled over, then you should consider checking this option as well. It should almost be a **S**tandard **O**perating **P**rocedure within your organization. I have found that most users wish to have everything, except for uncompleted appointments, rolled over. I have seen, in cases where activities have not been rolled over, that the user also tends not to complete activities, and the calendar table becomes bloated. Remember that the intended use of the calendar is to schedule as well as to complete activities creating historical archives of information concerning any given customer on their record. To not follow this paradigm would lessen the effectiveness of GoldMine within your organization.

This grouping has its own section in the UserID.ini, and depending on the settings that section might look like this:

```
[ActvObj]
AutoForward=1110112
AFStampSync=1
```

In the AutoForward statement, you will notice that there is a single 0. This represents the standard setting of everything checked except Appointments, as I have in my GoldMine Options, and as shown in Figure 3-20. I am perplexed as to why FrontRange would use a **2** for a flag on this option as opposed to the standard **0**/**1** (Off/On) bit switch.

I wonder if any of my previous readers noticed that there is a new Auto Roll-over item in GoldMine Premium 8.5x? That's right, it was the addition of the ☐ **Alarms** (no hot key) option item. Whereas, in the past, Alarms would be automatically rolled over with the activity, one now has the option to determine if, in fact, they wish the Alarms to be rolled over independent of its activity.

Figure 3-21

Again, I have taken the next frame from Figure 3-7, untitled, and I am showing it here in Figure 3-21. This frame has a single item included within it. The **Calendar refresh rate (sec):** doesn't affect the user as much as this setting will impact the network traffic. The default setting for this item is 0. This setting instructs GoldMine to disable the refreshing of the calendar. This means that GoldMine will only poll the CAL table for items on this calendar when the calendar is first opened. In order for the user to see any new items that were added while the calendar was opened, the user choosing this setting, must close the calendar, and then reopen the calendar again.

A user could up this setting to 10 seconds which is the minimum setting other than 0 that GoldMine will permit. Doing this would cause GoldMine to poll the CAL table for changes that affect this calendar, every 10 seconds, and would refresh the calendar accordingly. This may be nice for the user, but could play havoc with network traffic, if this were an office with a large number of users all having similar settings. Most offices will set the standard for this setting, and I have found that most offices do not touch this setting once set. However, should your users wish to adjust this setting, they may do so in one second increments. Setting any number, 1 thru 9, will cause GoldMine to notify you, and set the minimum value of 10 seconds. Whenever anyone is thinking of adjusting this setting, they should make the decision in conjunction with their network administrator. I strongly recommend a User Override for this option.

The default setting is stored in the UserID.ini as:

```
[CalObj]
RefreshRate=0
```

That concludes my discussion of this, the **Calendar** tab, and brings us right into the **Schedule** tab as shown here in Figure 3-22.

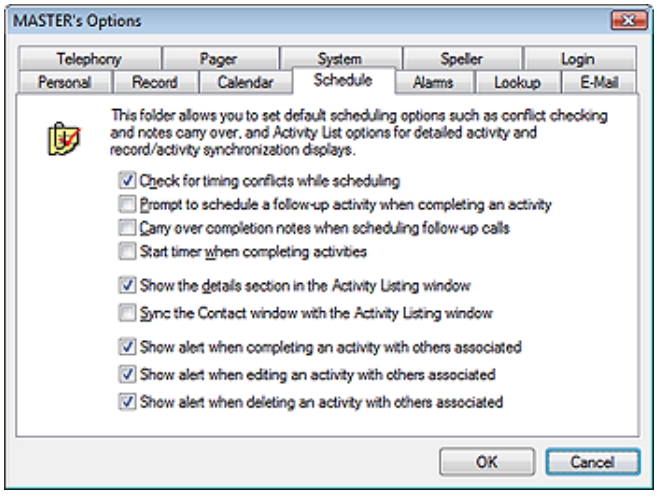


Figure 3-22

According to the dialog form, *"This folder allows you to set default scheduling options such as conflict checking and notes carry over, and Activity List options for detailed activity and record/activity synchronization displays."* This tab lets the user select/deselect 9 options now. Let's examine these nine options in a bit more detail.

Schedule

Note

I want to apologize for the graphics change here as we have now taken GoldMine Premium 8.5.1.10 live which places it on Workstations using the Vista operating systems.

The grey backgrounds should now disappear from all of the graphics.

WARNING

*When completing an activity in GoldMine with a duration of 00:00:00, and with any value, active or not, in the Status Bar Timer, the activity **will** have the duration field populated with that timer value. In GoldMine Premium, it is important to remember to **Reset Timer** after each use.*

☑ **Check for timing conflicts while scheduling** causes GoldMine to check to see if anything is already scheduled for said user/resource, as the user is trying to schedule a new activity/resource. Actually conflict checking occurs as the user leaves the **Time:** field. If there is a conflict in timing, said user will be notified immediately via a popup dialog form, and will be asked to **Ignore Conflict** or to **Re-schedule** the conflicting activity. Without this option checked, the user will be allowed to schedule multiple activities on the same date during the same time interval. By default, GoldMine has this option selected.

☐ **Prompt to schedule a follow-up activity when completing an activity** is the new GUI option for this version & build of GoldMine Premium, although I have discussed this in the GUIless section of previous editions of this book. Here is the section from my last book:

■ **Scheduled Activities - Automatically default to ☑ Schedule a Follow-up**

When completing an activity, this option is not selected in the default GoldMine configuration. If, on the majority of activities, the user should be scheduling other activities as a follow-up to the current activity, then you may want to have this option selected as a default condition for your users. To have the ☑ **Schedule a Follow-up:** checked by default, this would be the setting in the UserID.ini:

```
[GoldMine]
Followup=1
```

Naturally, with this option now within the GUI, I will no longer discuss this in the GUIless section of this chapter, however, it is important that you realize the functionality of said switch.

☐ **Carry over completion notes when scheduling follow-up calls**, and I want to bring your attention to the fact that GoldMine has stated **follow-up calls**. This option actually pertains to all scheduled activities, and not just to calls. The user must actually follow certain guidelines to have this option function properly for them. If the user checks this option, then as this user is completing a scheduled activity, if the user has notes in that activity, and if the user schedules another activity from this dialog form, then those notes will be brought forward to the new activity dialog form. At this point the user would just need to fill in the specifics of the activity, as all of the notes from the previously completed activity will be there, and will not require retyping. I am not sure why GoldMine has not checked this by default, but I feel that this is an extremely useful functionality to have in GoldMine. Personally, I always have this option selected for myself.

WARNING

*When completing an activity in GoldMine with a duration of 00:00:00, and with any value, active or not, in the Status Bar Timer, the activity **will** have the duration field populated with that timer value. In GoldMine Premium, it is important to remember to **Reset Timer** after each use.*

☐ **Start timer when completing activities** is an option that was placed here for the telemarketing powerhouse organizations. This option is one that probably should pertain to just calls, however, its selection affects all activities. Since its original intent was for the powerhouse telemarketers, lets look at that first. As a telemarketer, the more calls that you can place, the better off your position. One must make x number of calls per hour. The telemarketer prepares to complete the call, hence GoldMine automatically starts the timer. Now the telemarketer finishes the call, and, in turn, finishes completing the activity. GoldMine will enter the time, from the timer, into the **Duration:** field for the user. The key steps to remember, regardless of the activity type, is that the user must begin the process of completing the activity to start the timer, and when the activity is actually completed by the users clicking upon the **OK** button, then, and only then, will the timer be stopped. In the past this only applied to GoldMine completed calls. As of GoldMine 7.00.51018 this feature is applied to all activities that are being completed with a duration of 00:00:00, and where there is a value in the timer field. Immediately upon clicking the **OK** button, the timer would stop, and would not start new until they display the **Complete a Call** dialog form for the next call that they are in the process of making. Pow, Pow, Pow the telemarketer concentrates on the call, and powers right through their call listing for that day.

Recommendation

As you may have realized from my statements, in my opinion there are many disadvantages to using GoldMine Premium in the tabbed dialog form mode. Personally, I have separate dialog forms strategically displayed on my screen. I have the Contact dialog form directly above the E-mail Center splitting the vertical space evenly in half with their horizontal widths only taking up half of the horizontal space. Top to bottom, and taking up the remaining horizontal half of the screen, is my Calendar. I have found, over the years, that this layout works best for my usage of GoldMine Premium as opposed to the default tabbed configuration.

Well, if this option is activated, it affects all activities in GoldMine Premium. Would the user want the clock started as they are completing a Next Action? Probably not. Would the user want the clock started as they completing a scheduled Appointment, probably not. I guess that is why these are called users **Options**. It is up to the user to determine if selecting this option would be the correct choice in their usage of GoldMine. I have found, however, that very few users select this option.

☑ **Show the details section in the Activity Listing window** is an option that is selected by default, and appears to be a legacy hold over. Beginning with GoldMine Premium 8.5, this option appears to be completely irrelevant as the Activity Listing window now comes empowered with the new **Preview** toolbar.

☐ **Sync the Contact window with the Activity Listing window** is by default not selected. However, if selected, as the user highlights an activity in the Activity Listing window the Contact window will automatically synchronize to the associated company/contact record. This is not a critical feature if using GoldMine Premium in the tab mode, at least not in my opinion. If one has the screen real estate to show multiple windows, then this could be very advantageous, and will provide the user all of the information about the highlighted activity.

You now see another grouping of options with which the older version GoldMine users may not be familiar. GoldMine has no hot keys for these three options. Additionally, I would add that these options are not mentioned in the GoldMine Help files (weren't there in the GoldMine 7 Help files either), therefore one would have to speculate that the option description is self-defining. One could interpret these option statements improperly however. In general terms, these options are referring to Recurring activities that have been scheduled at the same time. These UserID.ini warnings have been in use for some time now, however, GoldMine 7 was the first version to make them accessible via the GUI.

☑ **Show alert when completing an activity with others associated**
☑ **Show alert when editing an activity with others associated**
☑ **Show alert when deleting an activity with others associated**

In my UserID.ini, these options on this dialog form would be represented as:

```
[ActvObj]
ConflictOn=1
ShowDetail=1

[GoldMine]
SyncActvObj=1
FURef=1
CompTimer=1

[Warning]
WarnAboutCompleteMultiLinkActiv=0
WarnAboutDeleteMultiLinkActiv=1
WarnAboutEditMultiLinkActiv=1
```

That concludes my examination of the Schedule tab.

Alarms

Next I will consider the options associated with the **Alarms** tab, Figure 3-23. Even though there is no frame around them, the user must first decide whether they want to have alarms or not, and if so, which type of notification they would like to receive. This option is represented by the three radio button option at the top of the dialog form. The first option, is, of course, to ○ **Disable alarms**, and I have found that a lot of users would like to select this option to avoid the annoying alarms. This does seem to defeat the purpose for having alarms in the first place. The alarms act as a tickler to remind the user that they have a task to accomplish, and it is the proper completion of this task that makes GoldMine the valuable tool that it is. The historical information on a company is valuable information to have when attempting to close a deal. Additionally, the activity may have been scheduled for them by another user, and/or an Automated Process. The user wouldn't necessarily be aware of the activity if it were not alarmed, and the user had not disabled their alarms. I think that the corporate standard should be to not select this option (see sidebar Recommendation).

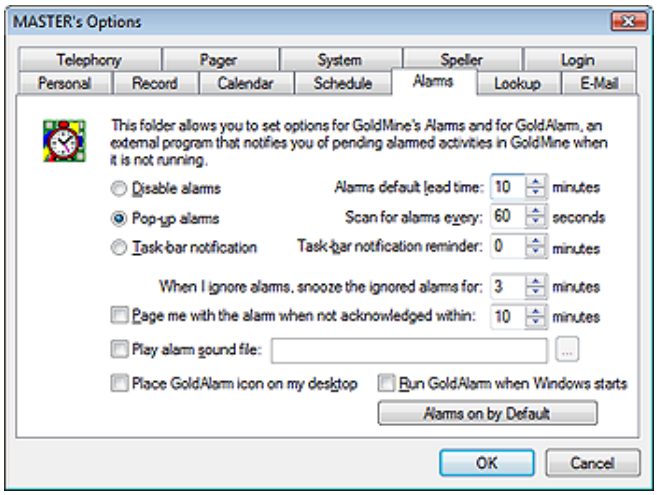


Figure 3-23

Recommendation

*In my opinion, a wise GoldMine Administrator would employ a **User Override** so that their users will have ○ **Disable alarms** disabled at the beginning of that users GoldMine session, at least, in its default state. I will discuss the User Override later in this chapter.*

The next radio button is the default selection for this option and it is to have ◉ **Pop-up alarms**. A pop-up alarm is another in your face dialog form to which the user must react. See Figure 3-24 on the next page for a sample of this type of dialog form. This type of pop-up dialog form is always on top of all windows display on your screen regardless of the applications that you have currently running. If the user were to try switching to another application, Ctrl-Tab, this alarm would still remain as a top level form.

The third selection for this option is to have a ○ **Task-bar notification** which is definitely not as in your face to the user as a pop-up alarm dialog form. This type of alarm can be easily missed by the user, see Figure 3-25 on the next page. The user should receive the default audible beep along with this type of notification (a sound file may be played in place of the default beep, I will discuss this option shortly), however, if the user is working at 1024x768 screen resolution or better, this type of notification could easily be overlooked. For criminy sakes, you can barely notice it here in this book.

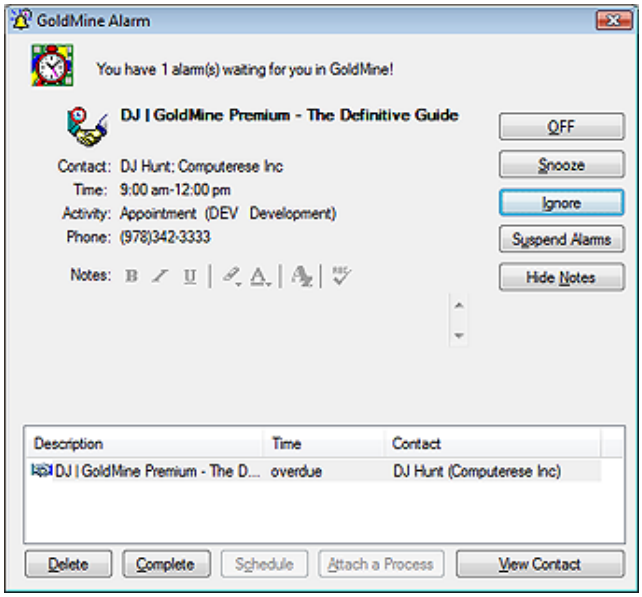


Figure 3-24

How is any user supposed to notice that in their Status Bar while they are multitasking is beyond me. I don't think that is going to happen, and I certainly do not recommend this option choice.

When the user is scheduling an activity, and they check to have an alarm, GoldMine will automatically subtract the lead time specified in the **Alarms default lead time:** box from the **Time:** set when the user is scheduling an activity. The lead time is specified in **minutes**. Later, in the **GUIless INI Statements** section of this chapter, I will discuss how to set up the UserID.ini to have GoldMine automatically check the Alarm option on the scheduling dialog form when the user is scheduling their activities.

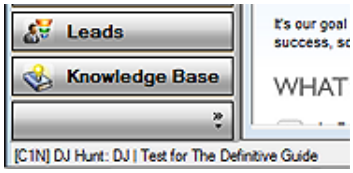


Figure 3-25

The next option that the user must consider is to **Scan for alarms every: xx seconds**, the default being scanning for alarms every **60** seconds. This is probably an optimal setting for this option. Again, you can see that there is the issue of network traffic. Every time GoldMine scans for alarms, GoldMine is creating network traffic. If you have 200 GoldMine network users, this could result in a substantial volume of network traffic. Users should consult with their Network Administrator before changing this setting. Let your Network Administrator determine what the optimal setting is for your corporate network load.

The next setting that the user must consider is the **Task-bar notification reminder: xx minutes**. Should the user select ○ **Task-bar notification**, which I had discussed earlier, there is no user interaction when the alarm is triggered. The notification could be easily overlooked as I had mentioned earlier. To minimize this possibility, GoldMine has given the user this setting to remind him/her, at a self specified interval, in the task-bar (aka **Status Bar**) until such time as the user properly completes this alarmed activity.

The GoldMine Pop-Up alarm does allow for user interaction, as you can see in Figure 3-24 above. One of those interactions is **Ignore** the alarm. Once the user selects to ignore the alarm, GoldMine only wants to allow it to be ignored for a predetermined length of time before it again pops up in the face of the user. The user, of course, determines how long a period of time must pass before the alarm is activated again. **When I ignore alarms, snooze the ignored alarms for: xx minutes**, is the option used to set this interval. The default snooze duration is 3 minutes. The user may opt to increase the amount of snooze time, or decrease the amount of snooze time. In most cases, if the user adjusts this time at all, the user opts to increase the snooze duration.

This next option works in conjunction with another collection of settings, which I have yet to discuss, the Pager settings. I will discuss those settings later in this chapter under the **Pager** tab options. For now, however, the next option under the **Alarms** tab is to ☐ **Page me with the alarm when not acknowledged within: xx minutes**. The default duration for this option is 10 minutes. This is fairly self-explanatory, yet it does instruct GoldMine to page the user if the user has not responded to an alarm within the 10 minute portal, in the default case, after the alarm has been tripped.

GoldMine next permits the user to identify a sound (**.wav**) file to be played in lieu of the default ding.wav that is played by GoldMine as an alarm is tripped. I have seen some users get very inventive with this option. The ☐ **Play alarm sound file:**, when selected, allows the user to browse their environment for a .wav file that they would like to have played as the alarm is triggered.

The UserID.ini settings for the **Alarms** tab as I have discussed to here could be:

```
[GoldMine]
AlarmFreq=60

[ActvObj]
AlarmsOn=1
AlarmsLead=10
```

Note

*Task-bar is actually a misnomer that has been carried through in many versions of GoldMine. This area, in most applications, is normally known as the **Status Bar**.*

Recommendation

*This is yet another good case for a **User Override**. The GoldMine Administrator should do everything in their power to prevent the user from defeating the GoldMine Alarm system. If I had my way these would not even be user options.*

Note

Although GoldMine will allow you to select an mp3 file, apparently GoldMine Premium doesn't know how to actually play David Bowie tunes.

WARNING

User Caveat: *If* ***GoldAlarm*** *is running, and the user starts GoldMine, GoldAlarm will close and the icon will no longer be available in the system tray. This is the expected behavior. However, upon closing their GoldMine session, the user might expect GoldAlarm to re-initiate itself, and it will not. If, after running GoldMine, the user wishes to reactivate GoldAlarm, they must do so by employing the desktop icon, if one was so created, or by selecting GoldAlarm from* ***Programs*** | ***Startup****.*

Note

Should the individual set ***OnByDefault*** *for* ***M****, messages, they must be aware that this setting is only applicable to the* ***GoldMine Internal E-mail*** *message, and does not apply to Internet E-mails.*

Note

Even though GoldMine lists the RecType of ***D = To-do****, the astute reader will realize that there is no option for an alarm when scheduling a To-do manually.*

Lookup

```
[GMAlarm]
PageAlarm=1
PageAfter=10
IgnoreSnooze=3
TrayReminder=5
PlaySound=C:\WINNT\Media\notify.wav
```

You now have two more options that go hand-in-hand. ☐ **Place GoldAlarm icon on my desktop** and ☐ **Run GoldAlarm when Windows starts**. GoldMine, when running, takes one license per user, away from the license pool. Contrary to my previous writings, running **GoldAlarm** in the **Silient Mode** does utilize a GoldMine license. The access to this application is through the use of the silent mode switch in the command line start up of GoldMine. For example, that command line, in the icons target field, might look like:

```
G:\GoldMine\GMW.exe /u:DJ /p:Password /s:GMAlarm
```

If the user had selected one or both of these options. There is no UserID.ini setting modified by the selection of either of these options. Their selection of either of these options immediately results in the icon being created on the desktop, and/or an icon being added to the Start-up folder under the Programs section of Windows. If the user selects to ☐ **Run GoldAlarm when Windows starts**, an icon  (new icon for GoldMine Premium) will show in the system tray indicating that GoldAlarm is running. This same icon in the system tray will result if the user starts GoldAlarm employing the desktop icon.

FrontRange has taken the liberty of moving another of the GUIless entries into the GUI realm. Those of you who are not new to GoldMine will notice the new **Alarms on by Default** button. Clicking on this button will provide the user with the GUI shown here in Figure 3-26. Let me pull this information from my previous version of this book for your review.

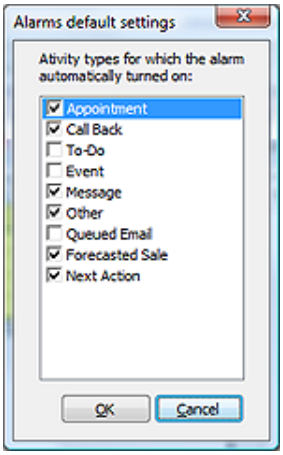


Figure 3-26

■ Scheduled Activities - Alarm automatically selected when scheduling activities

For instance, let's say that the user wants to have the alarm box, ☑ Alarm, automatically selected whenever they schedule an Appointment, Call, Next Action or a Sale. In order to do this, they would need to add to their UserID.ini, in the section shown:

```
[GMAlarm]
OnByDefault=ACTS
```

The user may select any activity RecType to have that activity alarmed by default. GoldMine uses the following RecTypes:

A = Appointment
C = Call
T = Next Action
D = To-do (even though there is no manual check box, this does appear to alarm To-Do's)
M = Message
S = Sales Potential
O = Other
L = Literature Request

My personal settings for this statement are:

```
[GMAlarm]
OnByDefault=ACTSOMDL
```

The reader is reminded that not all RecTypes will be alarmed by default in all versions of GoldMine. Each version of GoldMine has its own idiosyncrasies, so you will need to run a thorough test against your version.

Although one may still modify these settings by hand, FrontRange has now brought a GUI to these Options. Once you click on the new Alarms on by Default to bring up the dialog form shown in Figure 3-26, it is just a simple matter of your checking/unchecking those activities that you want/do not want alarmed by default respectively. As always, once you have made your choices, simply click on the **OK** button.

Now I want to move on to the **Lookup** tab. Under this tab, Figure 3-27 on the next page, you have a number of options that directly act upon the **Contact Search Center** dialog form within GoldMine. Whenever the user clicks on the **Search** icon on the GoldMine Toolbar, or selects one of the

Note

Please notice that I did not say, ...or simply double-clicks on one of the ***indexed*** *fields labels. You may now Search on any Contact1/Contact2 field by simply double-clicking on its label. Sweet!*

Note

New, in GoldMine Premium 9.0.2.36, is the ***More Options...*** *button. This build was released after this chapter was finished, hence, this sidebar note.*

New is the ability to change the ***Contact Search Center****,* ***Search by:*** *drop list of searchable fields. For instance: You may now add* ***Details*** *to your* ***Contact Search Center****,* ***Search by:*** *drop list.*

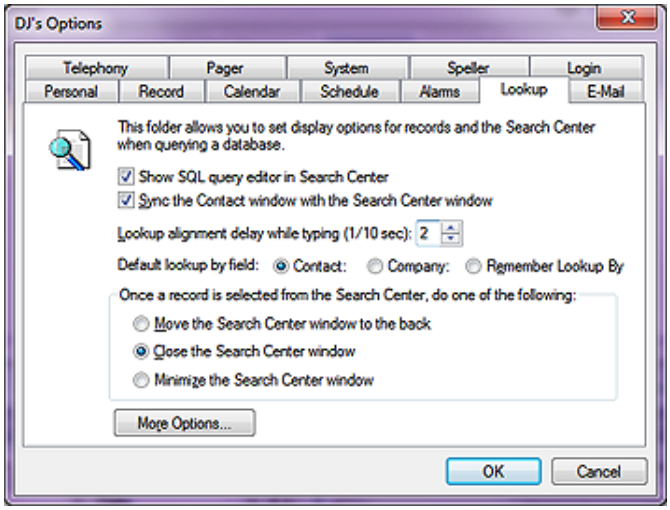


Figure 3-27

choices under the **Go To | Search ► | Contact Search Center** selection from the GoldMine menu, or simply double-clicks on one of the labels, this activity will bring up the Contact Search Center dialog form. Setting the options under this tab will control some of the attributes of that dialog form.

The first option is brand spanking new, and allows for a rather neat display of information. It is to ☐ **Show SQL query editor in Search Center** which is by default not selected.

I want to show you what the results of choosing this option might reveal. Take a look at Figure 3-28 (I had to cut the image for better presentation in this book).

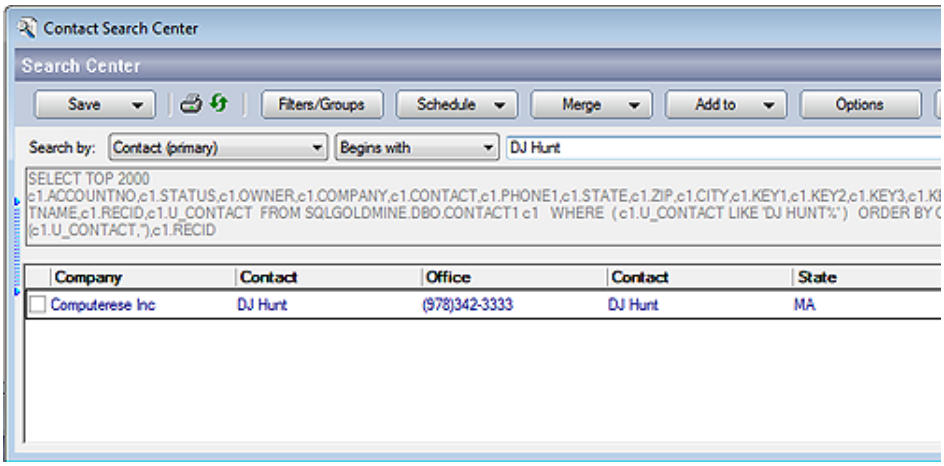


Figure 3-28

This is great. The end user gets to see the SQL Query that GoldMine intends to utilize to achieve their goal, and you can modify the query by clicking on the **Edit** button (not displayed in Figure 3-28).

Look, here is the query shown above:

```
SELECT TOP 2000 c1.ACCOUNTNO,
  c1.STATUS,
  c1.OWNER,
  c1.COMPANY,
  c1.CONTACT,
  c1.PHONE1,
  c1.STATE,
  c1.ZIP,
  c1.CITY,
  c1.KEY1,
  c1.KEY2,
  c1.KEY3,
  c1.KEY4,
  c1.KEY5,
  c1.LASTNAME,
  c1.RECID,
  c1.U_CONTACT
FROM SQLGOLDMINE.DBO.CONTACT1 c1
WHERE  ( c1.U_CONTACT LIKE 'DJ HUNT%' )
ORDER BY COALESCE(c1.U_CONTACT,''),
  c1.RECID
```

This is a nice way of easing your end users into understanding SQL Queries. Do you know what that coalesce() function is doing. I'll bet you will after seeing it used, and having your curiosity peaked.

Recommendation

*The ☐ **Sync the Contact window with the Search Center window** losses all of its value when GoldMine Premium is utilized in its new default Tabbed dialog mode. It would be my recommendation that your users utilize strategically placed dialog screens as has been utilized in previous versions of GoldMine.*

Recommendation

*I always recommend that an organization use a User Override to assure that this is set at the corporate level, selected, and not set as an individual preference. User Overrides are discussed later in the **GM.ini** section of this chapter.*

Note

Remember that you can search on any Contact1/Contact1 field by simply double-clicking on that fields label.

Note

I would like to point out that there is no particular order in which the instruction must appear in the UserID.ini as long as the instructions appear under the appropriate section headings. In fact, each time that a user saves these settings from within GoldMine, they may appear in a different order within the UserID.ini.

Alright then, let's move forward to where you are presented with is whether to ☐ **Sync the Contact window with the Search Center window** or not. By default, the **Contact** record does not move to the highlighted record in the **Contact Search Center** dialog form when the user selects a contact from the listing on that dialog form. The **Contact** record will remain at the last contact that the user was viewing prior to their having brought up the **Contact Search Center** dialog form. As the **Contact Search Center** dialog form only displays a few pieces of information about each contact, the user may find that not enough information is presented for them to be able to determine if, in fact, they have the correct contact highlighted. One way for them to view more information is to have the **Contact** record synchronize to the highlighted **Contact Search Center** record. This would allow the user to view all of the pertinent **Contact** record information, and they would, positively, know that they had found the correct contact. If this is the functionality that the user would desire, then the user would need to select this option.

The next option of interest is **Lookup alignment delay while typing (1/10 sec): xx**. The default value for this option is 2/10ths of a second. When typing in the information that they are searching for in the **Contact Search Center** dialog form, should they stop typing for 2/10ths of a second, GoldMine will execute a fetch call to the SQL database, and retrieve all of the records that meet the criterion specified at that instant with the first match highlighted, and Tagged (☑). The user may wish to change the delay while typing, by increasing or decreasing this value. The GoldMine developers have determined that 2/10ths of a second is an optimal delay when working in the SQL database environment. Less than this, and the network traffic and display results could be impeded, while more could be annoying to most your users. However, GoldMine still allows the user to adjust this delay to their own comfort level. If you are also the Network Administration, you may want to look at my recommendation in the sidebar if your network traffic is taking a hit.

The next user option was a direct result of user requests to the GoldMine/FrontRange organization, and is only applicable when the user brings up the **Contact Search Center** dialog form by clicking on the Toolbar **Search** button. In earlier versions of GoldMine, when the user clicked on the **Find** icon, as it was called then, the **Contact Listing** dialog form would pop up with a **Search by:** order of **Company**. Users inundated FrontRange with requests to make the **Search by:** order default to **Contact**. GoldMine made the change. Low and behold, the complaints again poured into FrontRange. Now, instead of trying to force everyone to one standard, FrontRange has chosen to give the users their choice as to which **Search by:** order it should default to when they, the user, click on the **Search** button. The **Default lookup by field:** ◉ **Contact:** ○ **Company:** ○ **Remember Lookup By** is default set for the contact lookup order, and this is where the user would exercise their choice if they needed to change from the default setting. You old time GoldMine users will notice a change from the past. ○ **Remember Lookup By** is a new addition to the available Options in the Lookup tab of the Options settings as of GoldMine Premium. I don't believe that I should need to explain what the selection of this option will do as it is fairly self explanatory.

The last option on this tab is another radio button selection. As you should be aware by now, GoldMine has one main window, and many, many other windows within that one window. In GoldMine Premium these windows may be kept as windows or the user may opt for the default Tabbed mode. If you are not using the new Tabbed dialog screen layout, then these windows can tend to clutter the main window. The first two possible option choices are applicable when using the Tabbed or Dialog Screen configuration for GoldMine. The third option, however, is not applicable if you are in the Tabbed mode, and is ignored if selected. Assuming that you are not in the Tabbed mode, the **Contact Search Center** dialog form, when activated, pops up as the top level form (over all other active dialog forms). The user must determine what happens to that search center window once they have completed their search. It is here that they make their choice.

Once a record is selected from the **Contact Search Center**, do one of the following:

- ◉ **Move the Search Center window to the back**
- ○ **Close the Search Center window**
- ○ **Minimize the Search Center window**

I believe that these choices are pretty much self-explanatory, and that they do not require any further discussion on my part for this book.

Now that I have concluded my discussion of the **Lookup** tab, I should show you the UserID.ini section, and the instructions that are affected by any changes on the **Lookup** tab.

```
[GM_SEARCH_CENTER]
DefField=0
SyncDelay=2
LookByShrink=0
SelectAction=0
SyncContact=1
ShowQueryEditor=1
```

E-mail

Note

*The **E-mail** tab has probably received the most revamping of any of the **Options** tabs with the release of GoldMine Premium 8.5.0.*

Many of these new features contain no information in the GoldMine Help files, and FrontRange has no documentation as to how to utilize these option settings. I do not have Exchange Server upon which to test these settings so, basically, we're flying blind here people. I will note these sections as we get to them.

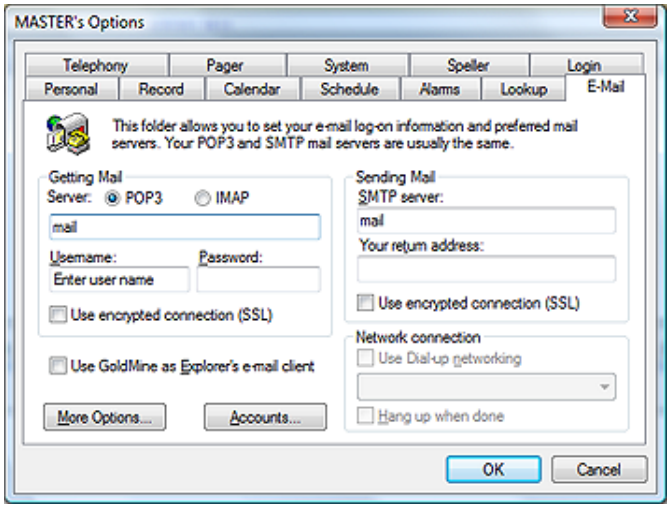


Figure 3-29

The **E-mail** tab is probably the most detailed tab, hence the most difficult to explain and to understand, as there are multiple layers of page frames, colloquially called tabs. This section of the chapter will take better than half of this chapter to explain, so get yourself a big cup of coffee, and hunker down for a long session. Let's start out with the basics first. GoldMine, in previous writings of this book, had only supported **POP3**/**SMTP** mail servers. As of 8.5.0, GoldMine is now capable of using **IMAP** to retrieve e-mail messages from the Mail Server. The **P**ost **O**ffice **P**rotocol version **3** (**POP3**) is the vehicle by which the user retrieves their e-mail, while the **S**imple **M**ail **T**ransport **P**rotocol (**SMTP**) is the vehicle through which the user sends their Internet e-mail. On the other hand, the **I**nternet **M**essage **A**ccess **P**rotocol (**IMAP**) is an application layer Internet protocol that allows the local client to access their e-mail on a remote server. The IMAP protocol is more appropriately utilized for retrieving messages from Exchange Servers without having to configure the Exchange Server with the POP3/SMPT protocols. I'm going to continue on for now, and I'll come back to this at a more appropriate spot in this chapter.

Note

*The ☐ **Use GoldMine as Explorer's e-mail client** option does not appear to function in my environment. I am utilizing:*

***Windows Vista Ultimate** (64 bit)*
***Office 2007** (**Outlook 2007**)*
Internet Explorer 8

On the first screen, under the **E-mail** tab, I am going to suggest that the user only worry about two items. The first of which is whether the user desires to ☐ **Use GoldMine as Explorer's e-mail client**. By default, Microsofts Internet Explorer uses Windows Mail as its default e-mail client, however, where I have Outlook 2007, my system utilizes Outlook as its default e-mail client. The user may desire to remain in a consistent environment when creating or retrieving Internet E-mail. If so, they may wish to select this option. Selecting this option tells Microsoft Internet Explorer to recognize GoldMine as the users Internet E-mail application (see Note in sidebar). The selection of this option makes no entry in the UserID.ini file.

Note

☐ ***Use Dial-up networking**, this option also affects GoldMine Synchronization. GoldMine Synchronization will try to use the dial-up network, if so selected, for doing its synchronization.*

The other section that a user should be concerned with on this tab is the **Network connection** frame. In this section, the user that is not attached to a network environment, may select to ☐ **Use Dial-up networking**. Come on now, does anyone really use this any more?. You may notice that my screen shot, Figure 3-29, has this section disabled. This indicates that I do not even have a modem configured for networking on my system. Users in a network environment do not usually have to concern themselves with this frame. Once selected, a drop-down list will be available listing all of the users configured dial-up accounts. The user should select the account that they want GoldMine to automatically employ when connecting to the Internet. Additionally, the ☐ **Hang up when done** option will be activated, and the user may also select this option.

Selecting these options could cause the UserID.ini to have these settings added:

```
[Internet]
UseDialUp=1
DialUpEntry=My Dialup Account
AutoHangUp=1
```

That's it for this page of the **E-mail** tab. From here, I would suggest that the user click on the **Accounts...** button which will bring up the dialog form shown in Figure 3-30 on the next page. It is from this dialog form that the user should manage their various accounts. You may establish quite a few accounts for number of different activities. Let's concern ourselves with the one account that is somewhat preset by GoldMine, and is designated by the green ball icon as the default account. The user may choose their own default account, as I have done, after they have added more than the one account. Let's start by having you highlight the one account that is shown, and then clicking on the **Edit...** button. This will bring up the dialog form shown on the next page in Figure 3-31.

This dialog form should be new to our old timers as well as our new readers. The first thing that I want the user to do is to give this account a descriptive **Account Name:** which will appear in the **E-mail Center** under the centers **Online** branch folder displayed in the navigation bar to the left. This name should be simple, yet clearly identify the account. As this account is currently set as the default account, it will appear in bold when viewed in the E-mail Center under the centers Online branch folder.

Note

Within GoldMine, it is possible for a user to set up as many as 256 different accounts from which they can retrieve/send Internet e-mail.

If you look at Figure 3-30, you'll notice that I, myself, have many accounts that I monitor. Some of these, I will actually be auto-retrieving such as my WebImport account, while others, my partner consultants, I will just be monitoring when they are not available.

Tip

*Should the user have the **Online** branch folder selected prior to entering their **Options** / **E-Mail** tab they will notice that the tab is disabled, and that they are unable to make modifications.*

*Insure that you are on the **Inbox** branch folder in the E-mail Center prior to attempting to modify your E-mail Options.*

☑ **Show the account in the E-mail Center** is disabled for this selection as this is the users default account. Should the user add more accounts, this option will become available on each account, and the user may then choose to show an account or not in the E-mail Center. Why would one setup an account, and then opt to not display it? The default account is always displayed in the E-mail Center, and this option will always be disabled for the default account. As the user is taking the time to set up the account, they would probably want to have the account available in the E-mail Center.

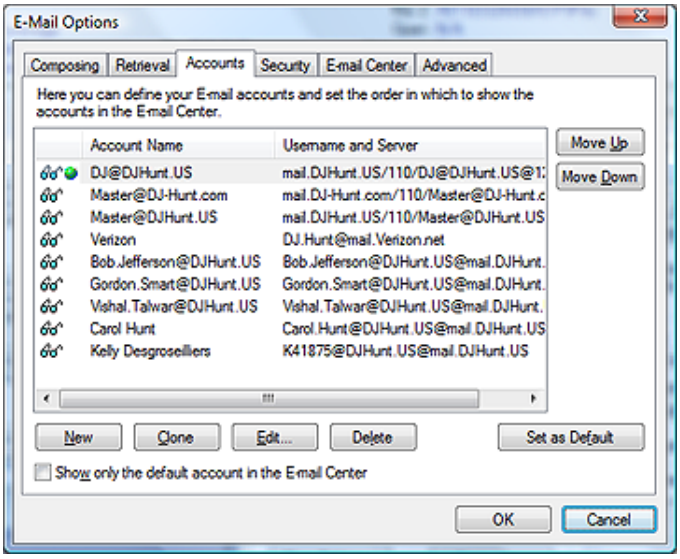


Figure 3-30

Next in turn is the **Getting Mail** frame, and here is where we go astray from our previous Options in prior builds of GoldMine Premium. The user is first required to identify type of Server from which they will be retrieving their e-mail, and, in this case, they may opt for either ◉ **POP3** (default) or the ○ **IMAP** server. Now they must identify the **Server:** name for this account. GoldMine will accept this as either a name or as an IP address. Users who do not possess this information, may acquire the necessary information from their GoldMine Administrator, their Network Administrator or their **I**nternet **S**ervice **P**rovider (**ISP**). A typical name construct would be similar to mail.DJHunt.US, while the IP address syntax might look like 216.149.153.109 or, in my case as I use K9 for spam control, 127.0.0.1. Again, either is acceptable in this field as long as it is a valid entry.

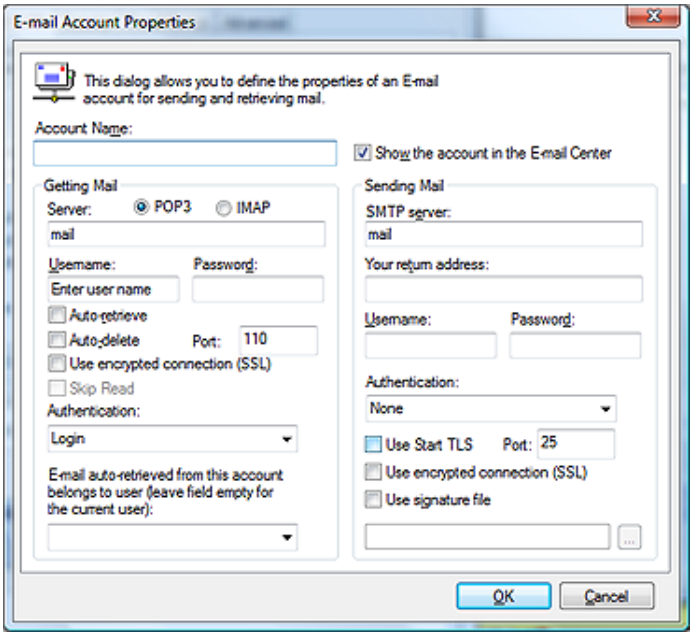


Figure 3-31

Tip

***K9** is a Shareware utility for Spam control that works well with GoldMine POP3 configuration, or any POP3 client configuration for that matter.*

*You may download it for free at **Keir.net**. I liked it so much that I made a $30.00 donation to their cause some 7 years ago when I first started using it with GoldMine.*

Tip

Should the user fail to connect to their account after having set up the account, the user should first try to re-enter their password. I have found that, because of the lack of password confirmation, the user frequently has a typo in their password, and careful retyping usually corrects the matter.

The next required field in the Getting Mail frame is the **Username:** field. Here, the user is expected to enter the user name that is required when logging into this account on the Server. This is normally only a user name, and not the users entire e-mail address, although it very well could be the entire e-mail address. An example of the user name, in one particular case that comes to mind, is DJ, while another, for my K9 configuration, was mail.DJHunt.US/110/DJ@DJHunt.US. The user would need to have this information, which may come from the Network Administrator, the GoldMine Administrator, or the users ISP.

Along with the username, there is usually a required **Password:** which must be entered in order to log into the account to retrieve e-mail. As the user types the password, only asterisks will be displayed. Additionally, there is no password confirmation field, such that the user should be very careful and very conscious of what they are typing into the password field. Remember that passwords are usually case sensitive.

Next, we see that there are two familiar options, ☐ **Auto-retrieve** and ☐ **Auto-delete**. These two options work hand-in-hand with each other. If the ☐ **Auto-retrieve** option is selected, GoldMine will attempt to log into the users account, at a pre defined interval (refer to the **Retrieval** tab discussed

Tip

When a user starts to complain about receiving the same message over and over again, it is usually because the user has set their accounts to Auto-retrieve, but have failed to set the Auto-delete option.

*Should the user select the ☐ **Auto-retrieve** option, they would be wise to also select the ☐ **Auto-delete** option.*

later in this chapter), and will retrieve any e-mail that is on the POP3 mail server. GoldMine will retrieve e-mail for all set up accounts that have this option selected. Should the user select this option, but refrain from selecting the ☐ **Auto-delete** option, then a copy of e-mail will be retrieved from the e-mail server while the original e-mail message will remain on the POP3 server. GoldMine does not mark the retrieved messages, on the server, as having been read. It is important to understand this action. This particular setting could result in the same message being retrieved multiple times into the **E-mail Center**. Selecting the ☐ **Auto-delete** option will result in e-mail being removed from the **Internet Mail Server** upon successful retrieval into the GoldMine **E-mail Center**.

Within the Getting Mail frame now are some new options. I speak specifically of the **Port:** value, and the ☐ **Use encrypted connection (SSL)** option. Now the **Port:** value used to appear on another dialog form in prior builds of GoldMine, however, is given a more prominent focus in GoldMine Premium 8.5.0. As an example, I utilize port 9999 in my configuration for K9. As for the ☐ **Use encrypted connection (SSL)** this is what the GoldMine help files have to say about that: *Select this checkbox to use a secure connection. This ensures the privacy of the connection between your account and the server. Enabling SSL does not verify the identity of your account or the e-mail server, but does prevent a third party from listening or recording messages sent between them.*

If you have chosen to use the IMAP configuration, you will also be able to set individual account options for the ☐ **Skip Read** capability. This item used to be a POP3 general setting under the **Retrieval** tab, and still is quite frankly, however, IMAP permits the individual account configuration for this option. Not having the IMAP capability, I cannot test this out for you, however, I can assure you that selecting this option under the **Retrieval** tab for the POP3 configuration really had no effect.

And next is the **Authentication:** type droplist which varies depending on whether you have selected POP3 or IMAP. This frame allows the user to tell GoldMine which protocol to use for the account and password on the Internet mail server the POP3 or IMAP. Under POP3 your options are:

Login
NTLM

While IMAP offers the additional login type of:

CRAM-MD5

The proper selection of the authentication type is important, and although it is a user option, it is not determined by the user. The protocol the user must employ is determined by their Internet Service Provider. The user must acquire this information from their GoldMine Administrator, or, in the absence of a GoldMine Administrator, directly from their ISP or their Network administrator.

Tip

Many users desire the abiltiy to send e-mail messages using different return addresses. Here is how they would set up the various accounts to accomplish this feat.

*Set up one **Account** for each desired return e-mail address. The **Getting Mail** side may be the same or different for each account, it matters not. (If the **Getting Mail** side is the same for multiple accounts, only the first account needs to be considered for **Auto-retrieve** and **Auto-delete**).*

*However, on the **Sending Mail** side, in **Your return address:** they should be set up differently for each desired return address. I strongly recommend that the return e-mail address be entered in a format similar to:*

DJ Hunt <DJ@DJHunt.US>
Carol Hunt <Carol.Hunt@DJHunt.US>

*Now the proper name, and the e-mail address will appear in the **From:** field of the outgoing e-mail when selected. That's right, the user must select the correct **From:** account from the available drop list on the e-mail message. This cannot be done automatically.*

Now we come to: **E-mail auto retrieved from this account belongs to user (leave field empty for current user):**, and this too used to be elsewhere within the GoldMine Options. This permits users to configure an account, usually set for Auto-retrieve, for another user, for instance while they are away, so that the active user can retrieve the E-mails for said user, and have these E-mails retrieved to the away users account within GoldMine. There are many other uses for this section (WebImport comes quickly to mind), however, I leave this usage to your imagination. The associated drop list will display all of your configured UserIDs.

We now want to move ahead and discuss the **Sending Mail** frame, and again refer to the previously displayed Figure 3-31. In this frame the first field that the user encounters is the **SMTP server:** field. The **S**imple **M**ail **T**ransport **P**rotocol (**SMTP**) is the protocol by which the users e-mail is sent from within GoldMine. Most commonly, regardless of how many accounts the user has, they will only have one SMTP server (see Tip in sidebar). It is not atypical for an ISP to name the POP3 server and the SMTP server with the same name, but this does not hold true in all cases. Again, the user must acquire this information from their GoldMine Administrator, or, in the absence of a GoldMine Administrator, directly from their ISP or Network Administrator. In my case, for instance, the POP3 server is named mail.DJHunt.US while my SMTP server is named mail.authSMTP.com. Why the difference you might ask? Well, mail.authSMTP.com is a paid SMTP service which permits me to eBlast up to 10,000 e-mail messages per month (you may select from different plans), whereas, my Verizon account was limiting me to 100 messages per hour or a maximum of 500 messages per day. I found this to be too constraining for my needs. If you do a lot of eBlasting, you too may want to consider using one of the SMTP services like www.authSMTP.com.

In **Your return address:** field, the user would enter this accounts user e-mail address. As we have alluded to, the e-mail address is usually, but not always, a combination of the username and the domain name. A typical address could be: DJ@DJHunt.US. In case the user wants a different name associated with the e-mail address, the return address should be entered thusly:

DJ Hunt <DJ@DJHunt.US>

Now we are asked for two more pieces of information. GoldMine needs the SMTP **Username:** and **Password:** which for my authSMTP account would be **ac28103** and ********** respectively (you didn't really think that I was going to give you my password now did you?). Now you older GoldMine users will remember that all of this information was presented to you under the advanced options area of previous GoldMine Premium builds.

So we must now be concerned with SMTP **Authentication:**, and, unlike POP3/IMAP Authentication, this drop list is not affected by you selection of POP3 or IMAP. Here are your choices:

Auto
None
Login
Plain
NTLM
CRAM-MD5

Personally, authSMTP requires **Login** authentication so that is what I would select for my environment.

Okay, I have to be honest here, with nothing in the GoldMine Help files, I had to find out about the ☐ **Use Start TLS** option via a Google Search. Here is the results of my search.

The Transport Layer Security (TLS) protocol provides communications privacy over the Internet. The protocol allows client/server applications to communicate in a way that is designed to prevent eavesdropping, tampering, or message forgery.

If it is determined that you have a need for TLS then you will also need to define the TLS **Port:** which is defaulted to port 587. Should you need to use the TLS protocol, you will not be able to use the next option which is to ☐ **Use encrypted connection (SSL)**. It is one or the other, and not both protocols. This is the same protocol that you may have used for retrieving your E-mails in a secure manner. From the GoldMine Help files:

Choosing this option will also cause a switch to a dedicated secure connection port number on the SMTP server.

Lastly is an option selection to ☐ **Use signature file**. Now I want to be clear here, that the selection of this option pertains to the inclusion of a signature text file into all of the users outgoing e-mail messages utilizing this account. The words **text file** are the key words in that sentence. GoldMine Premium allows the option of using HTML e-mail. I will discuss that later, but, if the user plans to use HTML e-mail, then this option should not be selected. Even if selected, should the user be using HTML e-mail, this option will be negated. Today,almost everyone is utilizing HTML Templates, and I'm surprised to see that GoldMine is maintaining this legacy option.

Regardless, should the user decide to select this option, they would then be expected to point to the location of the signature text file. They could do this by typing in the fully qualified path and name of the text file, or by clicking on the ellipsis (**...**) button, and browsing to the file. The end result must be the fully designated path and file name of the signature text file. One could expect to see a name such as **G:\GoldMine\EMail Sig\Generic.txt** in this field.

If I look at this section of the UserID.ini, with this information included, I might see:

```
[Internet]
UseDialUp=0
POP3_Account=DJ@DJHunt.US
POP3_Server=127.0.0.1
POP3_User=mail.DJHunt.US/110/DJ@DJHunt.US
Return_Address=DJ Hunt <DJ@DJHunt.US>
SMTP_Server=mail.authSMTP.com
DelServerMail=1
AutoGetMail=1
SMTP_User=ac28103
POP3_Pass=02443B221D5845405B22BE8F0242236E69714548BA8B8D8F
POP3_Port=9999
SMTP_AuthMode=1
SMTP_Pass=0244606F1D5845405B22BE8F0240726B696AB687898B8D8F
IN_Server_Type=1
IN_OverSSL=0
OUT_OverSSL=1
SMTP_StartTLS=0
SMTP_Port=587
UseSigFile=1
SigFile=Y:\GoldMinePE\eSignature\Generic.txt
```

Note

You may read about the TLS protocol yourself by visiting:

ftp://ftp.isi.edu/in-notes/rfc2246.txt

Recommendation

*Use HTML templates for a more acceptable corporate image. Few people or organizations use Plain Text e-mails any more. Create a separate default **New**, **Reply**, and **Forwarding** e-mail template for each user signature required.*

Do not let the users create these independently as a corporation you would want to present a consistent and professional look to your clientele.

Note

*These settings, except for the **UseSigFile**, and **SigFile**, are my settings for using this account with the spam killer K9 (**www.keir.net**). I highly recommend this product as a spam killer for your **GoldMine E-mail Client**.*

Note

*Note that the variables **POP3_Pass** & **SMTP_Pass**, which contain the account password, is encrypted. Passwords may only be entered through the GUI in GoldMine, and must never be added to the UserID.ini directly.*

As the user adds subsequent accounts, this section will be appended to as many times as the user has accounts. The variable names will change to indicate the order of the account. For example, subsequent POP3 account variables will be named POP3_Account1, POP3_Account2, POP3_Account3, etcetera. This, then, pretty much sets up an Internet E-mail Account.

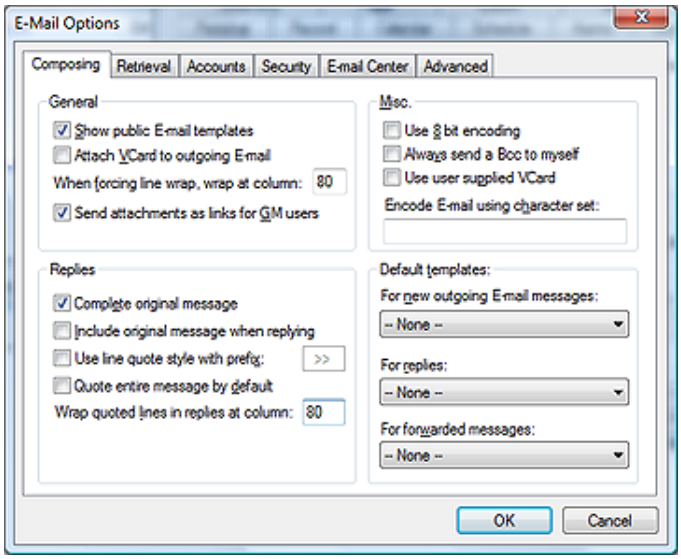


Figure 3-32

With this being accomplished, the user will have successfully set up one Internet e-mail account. They may now continue on and set up all of the additional accounts to which they have access. I would now ask you to refer back to Figure 3-29. From there, we would continue with the setting up of the options on the **E-Mail** tab. Notice that there is one last button that I have, as yet, not discussed, that being the **More Options...** button. Clicking upon this button will produce the dialog form as shown here in Figure 3-32.

The reader will immediately notice that we have encountered another dialog form containing 6 tabs from which the user may set many more Internet options. Actually, it is the same 6 tabs as shown in Figure 3-30, however, I am entering this dialog form from via a different avenue. On this, the **Composing** tab, alone, there are 4 different option frames. They are the **General**, **Replies**, **Misc.**, and the **Default templates:** frames.

I begin with the **General** frame, and the users first option is to ☑ **Show public E-mail templates**. As you can see, this option is selected in the default state. As a user adds e-mail templates in the **Document Management Center**, their templates are available for use by a user when composing a new Internet E-mail message for instance. Having this option selected means that, not only will this users templates be displayed on the list when composing an Internet E-mail, but also any templates that were created for the user group **(public)** will be displayed at the same time and be available for immediate use. As stated previously, at any point a user may access and use the templates of any other user from the template drop list. Why wouldn't you want the user to have the widest available selection of templates available, hence, this option is checked by default.

☐ **Attach VCard to outgoing E-mail** is an option that is not selected in the default state. Think of the VCard as a virtual business card (why not, that's what it is). It could be an attachment that is sent with every e-mail that the user sends out. It contains information from the **Personal** tab of the users **Options**, as well as information that is contained within the licensing of the GoldMine product. Here is a sample of the contents of a typical users VCard:

```
BEGIN:VCARD
VERSION:2.1
ORG:Computerese;Application Development
FN:DJ Hunt
N:Hunt;DJ
TITLE:Owner
ADR;WORK:;;150 Pratt Road;Fitchburg;MA;01420;USA
TEL;WORK;PREF:(978)342-3333
TEL;WORK:9783423333
TEL;FAX;PREF:
TEL;FAX:
EMAIL;PREF;INTERNET:DJ@DJHunt.US
END:VCARD
```

If the receiver has a system that is VCard capable, the system can decipher the information contained in the **.vcf** file. The deciphered information could then be easily, if not automatically, imported by the receiving application. GoldMine is VCard enabled, as is Microsoft Outlook. Each of these applications could use the VCard information to create a new Contact record. It is my experience, however, that few people take advantage of the VCard capabilities, hence negating its value. Should the user select this option, GoldMine will create a text based VCard, see above, for the user in the **...\Template\UserID** folder. For the card shown above, the path would be: **C:\ProgramData\GoldMine\Template\DJ\DJ.vcf**

Tip

*There are VCard creators available that create graphical VCards. Should the user decide to use one of these applications to create a VCard, they must put the resulting VCard in the **...\Template\UserID** folder. The file must be named with the UserID plus have the **.vcf** extension.*

*It is important, should the user select this approach, that they also select the option ☐ **Use user supplied VCard** which is in the **Misc.** frame of the **Composing** tab. This will prevent GoldMine from updating the VCard with its self contained user information.*

The next setting, however, is **When forcing line wrap, wrap at column: xxx**. When forcing line wrap is sort of ambiguous, GoldMine always forces line wrap. If the user doesn't change the default setting, then GoldMine will force the line wrap, in the body of an e-mail, on the 80th character typed on a line. This is a standard setting that many e-mail clients employ.

Old time GoldMine users may notice the absence of the ☐ **Use signature file:** option. FrontRange is pushing for the use of HTML E-mail Templates, and wants to get away from the Plain Text signature files. I will discuss HTML E-mail templates later in this chapter. I find that most organizations prefer the HTML capabilities anyway, and appearance that could result from the use of HTML templates.

Recommendation

*It is my opinion that you should **never** send **GoldMine E-mails** to **GoldMine Users**. You should always have a record for your employees in your GoldMine database, and you should always send them an **Internet E-mail** as opposed to GoldMine Internal E-mail eliminating the need for the use of this option all together.*

On the other hand, we still have the ☑ **Send attachments as links for GM users** option, which is the next, and the last option in the **General** frame, and, as in the past, it is selected by default. When the user composes an e-mail message that contains attachments, and then includes GoldMine users either as **cc:** or **bcc:**, if this option is selected, those GoldMine users will receive the path information to the attachment as opposed to receiving the attachment itself. It has been my experience that the selection of this option is the cause of an inordinate amount of frustration. How frustrating is it, for instance, when you receive an e-mail that has a link to an attachment in a path that is not accessible to you, the recipient? Even though having the attachment included as part of the Internet E-mail increases the size of the e-mail, in my opinion, it is better to have the attachment, than to not have access to the attachment at all. Naturally, these are user preferences, and the ultimate decision rests with the individual setting these preferences.

That concludes my discussion of the General frame. The resulting UserID.ini settings from this frame could look something like these:

```
[Internet]
LoadPublicTemplates=1
SendVCard=1
ForceWrapAt=80
GMAttachAsLinks=0
```

Continuing on, see Figure 3-32, I begin with a new frame that is called **Replies**. This frame has five options that control items that users can modify pertaining to their replies to an Internet E-mail.

The first option is selected by default, ☑ **Complete original message**. This option simply instructs GoldMine, when selected, and it is checked by default, to **Fast File** in History (ContHist/Mail-Box) the received message, after the user has relied to said message. No user interaction is required to Complete the incoming Internet E-mail message to send it to History, if this option is selected. The downside to this might be that there is no user interaction, therefore, the user can not add any Notes, Activity Code, or Result Code to the historical activity unless they edit the historical activity after it has been created. I have found, most often, that the user would not want to, nor need to, add notes to a historical activity for a received message, as the message, subject, and body usually provide satisfactory historical information. Therefore, most users will not change this default selection.

Recommendation

As this feature does not appear to work except in plain text mode (even that is questionable), and as I always recommend using HTML templates, I would recommend that the user not be allowed to select this option.

The option to ☐ **Include original message when replying** had been added in GoldMine version 6.00.21205, and is still retained today. The release notes for this feature stated (there is nothing in the GoldMine Help files even as of GoldMine Premium 8.5.0.89) that this option is for Outlook style original message quoting with color coded replies. This setting is independent of the ☐ **Quote entire message by default** option, and quoted replies will use a color coded cycle of blue, red and green. In my testing of this feature, it was found to only work if the user were not using templates, defined later, in the **Default templates:** frame under the Composing tab. This option has never worked properly, and is now not selected in its default state. With FrontRanges push to use HTML Templates I'm, quite frankly, surprised that FrontRange even left this option in the GUI.

The next item is an option, and a setting combination. The user must decide to ☐ **Use line quote style with prefix: >>** or not. If one leaves this option unselected, quoted text, in a reply, will be enclosed by opposing chevrons similar to this:

> This is the quoted text portion
> of the reply <

Where as, if the user selected this option with the default prefix, then each line of the quoted text would be prefaced with the prefix similar to this:

>> This is the quoted text portion
>> of the reply

The user may get as creative as they wish with the prefix, and I have seen some pretty elaborate ones in my day.

Note

*In my usage, I always select only those portions of the body that I want quoted in my reply, therefore, I do not choose to select the ☐ **Quote entire message by default** option.*

*It really and truly comes down to users usage, and the setting of the user **Options** to accommodate said usage.*

Note

*Because of the unusual representation of the wrap statement, I would like to emphasize its spelling: **WrarReplyAt**. I don't pretend to understand the reasoning behind this syntax, I only document it as it was unexpected.*

Note

*One thing that I have eluded to, but never stated (I believe) is that if one accepts a default option as is, then there may be no entry in the **UserID.ini**. For Instance: **QuoteAll=1** will appear in the UserID.ini, whereas, **QuoteAll=0** may not. Is it wrong if you include **QuoteAll=0** by hand? Certainly not, and it will function perfectly well.*

The next option could be an important one for the user. If the user selects to ☐ **Quote entire message by default**, then each time the user replies to an Internet E-mail message, if the user hasn't selected (highlighted) any text in the original message, everything in the body of that message will be requoted in the reply message with the designated prefix for each line. This could get quite confusing. Most often, I have seen that users do not select this option, and are very careful to select (highlight) only that portion of the message that they wish to have quoted in their Reply E-mail message.

The last setting in the **Replies** frame is similar to a setting that I discussed in the **General** frame. Here the user is instructing GoldMine as to where to wrap any quoted text in the reply message. The default setting is to **Wrap quoted lines in replies at column: 80**.

Making changes in the GUI to the **Replies** frame could result in the UserID.ini looking similar to:

```
[Internet]
CompleteOnReply=0
NewQuoteStyle=0
QuoteString=" >>"
QuoteAll=1
IncludeOriginalInReply=1
WrarReplyAt=80
```

The next frame is rather interesting, at least in the build that I am writing against. The frame is **Misc.**, and the observant reader will notice that frame has been given a hot key. I have no clue as to reasoning behind the hot key functionality, but it is there, none the less. Alt-M drives the selection cursor to this frame, but selects nothing. If there is some logic here, it has certainly eluded me.

The first option in this frame is to ☐ **Use 8 bit encoding**, and I won't even pretend to understand what selecting this option accomplishes. Instead I will supply the reader with the information contained in the GoldMine help file. *"**Use 8 bit encoding:** Converts 7-bit encoding into 8-bit encoding. Useful for international GoldMine users."* I don't know where or when this would be applied, only that it is available as an Option. There is a little hint there that it may be useful for my international readers so if any of you know how this helps you, please send and e-mail along to **DJ@DJHunt.US**.

Moving on, we have the next option which is to ☐ **Always send a Bcc to myself**. Selecting this option will cause GoldMine to send a Blind courtesy copy of the outgoing message to that users e-mail account. This is rather redundant as a copy of the message is saved to history on the contacts record, and in that users E-mail Center, however, it is a feature of which the user may want to take advantage. Just because I don't want to fill up my mailbox doesn't mean that another user would be of that mind set.

Note

In older versions of GoldMine, one could achieve this same result by creating their VCard independantly. Placing it in the correct folder under GoldMine, and making the file Read Only.

Next we have another option whose function is not at all clear, at least to me, and that is to ☐ **Use user supplied VCard**. Now, if I were just reading the option, and using that as my basis for a description, then I would assume that the user could create a VCard external to GoldMine and use that VCard without having GoldMine overwrite the file with its own VCard. That is assuming that the user selected this option. On the other hand, if I look at the information supplied in the GoldMine help file: *"**Use user supplied VCard:** Ensures the VCard attached to your outgoing e-mail includes updates."*, this leads me to a totally different understanding. My testing of this option seems to support the first explanation. If the user has created a VCard external to GoldMine, said VCard must still be named UserID.vcf (i.e. **DJ.vcf**), and must still reside in the **...\Templates\UserID** folder, however, with this option selected, it would appear that GoldMine will no longer try to create its own version of the VCard, and will not try to overwrite the users supplied VCard vcf file.

The last setting in the **Misc.** frame is to **Encode E-mail using character set:**, and it is up to the user to designate the character set to be used if they desire anything other than the default character set to be used for encoding. It would be important for the user to select a character set that is common to all systems for the decoding, of any so encoded message, to be able to function properly. For example, to use the Western European character code, type iso-8859-1. If left blank, GoldMine uses the local default character set for encoding.

Selecting all of the options in the **Misc.** frame could produce a UserID.ini section similar to:

```
[Internet]
Use8BitEncoding=1
BccToSelf=1
KeepUserVCard=1
MailCharSet=iso-8859-1
```

The last frame that the user encounters on the **Composing** tab is the **Default templates:** frame. Notice, again, that there is a hot key for this frame within GoldMine Premium. If the user has created E-mail Templates, or if the user has elected to show and use Public E-mail Templates, then the user may set, in these settings, those templates to be employed under each condition. The user can not

type into these setting areas, hence they must choose from the list of available templates. The circumstances for which the user may set the usage of a template, and their corresponding UserID.ini statement are:

For new outgoing E-mail messages:

DefaultTemplate=DMDSD7W*;!9UR#Y

For replies:

DefaultReplyTemplate=DMDSD7W*;!9UR#Y

For forwarded messages:

DefaultFwdTemplate=DMDSD7W*;!9UR#Y

To the right of the equal sign (**=**) in each of the UserID.ini statements is the **RecID** for the E-mail Template selected for that category. As this value is not encrypted, in theory, the UserID.ini could be modified by hand if the modifier was aware of the **RecID** for the particular template in question.

WARNING

*If the user is not using templates for any of these categories, then the template field for those not being employed **must** be set to **-- None --**. Having a blank in any of these fields means that GoldMine is trying to use a template with an empty title. This has been the cause of many a GoldMine Administrators nightmares.*

Note

*These items might be more easily established through the Document Management Center, now a dialog form simply named **Documents**. The user can simply right-click any template there, choosing the **Set as Default ► New Message** option, or either of the other two possibilities.*

The **Retrieval** tab is next in our order of progression, Figure 3-33. Under this tab we see two frames, and a separate settings area below these frames. The first frame that I will discuss is the **Retrieval Options** frame, and the first option in this frame is to ☑ **Open 'Read E-mail' dialog on retrieval**. This option is selected in the default state. The result of selecting this option will be, as the user downloads their e-mail messages, after the first message is retrieved, that first message will pop up immediately in the **Read E-mail** dialog form. If there are more messages being retrieved after the first message, then they will continue to be retrieved in the background while the user is reading, and processing the first message.

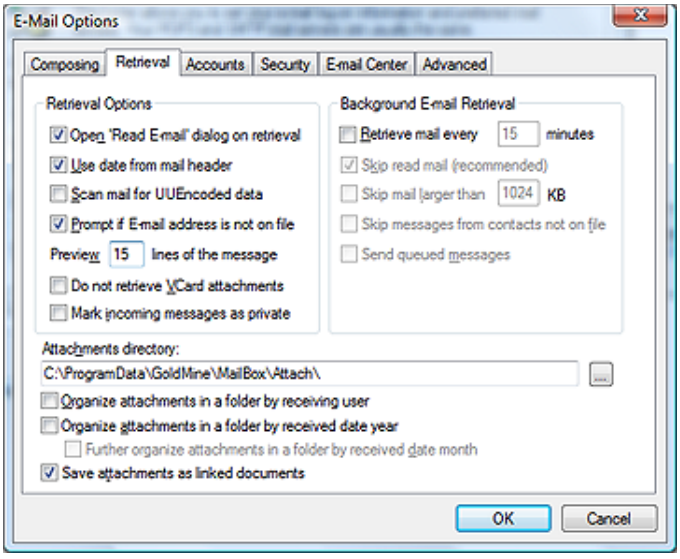


Figure 3-33

Recommendation

Should the user accept this default setting, they are advised to not select Messages in the Auto Roll-overs grouping under the Calendar tab.

☑ **Use date from mail header** is, again, selected in the default state. Any e-mail received, with this option selected, will have the GoldMine **Cal.OnDate** field stuffed with the sent date contained within the e-mail header information. This will mean that the e-mail messages will be posted to the users calendar, and presented in the **E-mail Center** in their true chronological order. If the user does not select this option, then all of the e-mail that is received will have the **Cal.Ondate** field stamped with the date on which the e-mail was retrieved.

The users next option is to ☐ **Scan mail for UUEncoded data**, and this option is not selected in the default configuration. Should the user select this option, it will have no affect on the retrieval of MIME encoded files, however, it will cause GoldMine to scan the Internet Mail Server for any e-mail messages that contain UUEncoded attachments. We're not sure why the GoldMine product would not ship with this as a default selected option as it does nothing but enhance the capabilities of GoldMine as an e-mail client.

In the next option, the user may choose to have GoldMine ☑ **Prompt if E-mail address is not on file**, and this option is selected in the default configuration state. Part of what is inherent in GoldMine is its ability to have all information about a contact linked to the contact record. On incoming e-mail, GoldMine, to function as it would like, will want to link the Internet e-mail message to a contact record. With this option selected, if GoldMine cannot find that e-mail address in its tables, GoldMine will prompt the user to create a new record, or to find an existing record to attach this e-mail message, as well as additional options. If the user does change the default for this option, GoldMine, if it can not find the e-mail address in its tables, will simply not link the message to any contact record. It would then be the users responsibility to link the message by hand to the appropriate contact record, or not, as the GoldMine Administrator dictates. I would mention that this option only functions when se-

lected, and the user is retrieving their e-mail interactively. If **Auto-retrieve** is selected, and retrieving the e-mail, no user prompt will be displayed, as there may be no user for which to display the dialog form. After all it is supposed to be automatic (as long as GoldMine is running).

In the next setting the user is establishing how many lines of an e-mail message are to be previewed in the **E-mail Center** preview window. This setting is rather nice as it allows the user to preview the e-mail message in the **E-mail Center** prior to even downloading it from the **Internet Mail Server** if they so choose. In the **Preview xx lines of the message** the user may choose to preview more or less than the default 15 lines. I have seen many users set this value up very high so that they can preview nearly every message in its entirety while it is still on the **Internet Mail Server**. It is not uncommon for users to set this number to 100 lines. In fact, I too have mine set to 100 lines. Aren't User Options great? We all get what makes us most productive.

In this same frame, the next option is, by default, not selected. This option, ☐ **Do not retrieve VCard attachments**, when not selected, means that GoldMine will not download any VCards as it downloads the corresponding Internet e-mail message. This means that, should the user need to create a new record with which to link the e-mail message, the VCard information will not be available for the creation of the new record. The user will be required to add the information for the new record by hand, typing the information into the record or copying & pasting the information from the e-mail itself. The downside to selecting this option is that the user could fill a hard drive with VCards that are no longer required. I'm glad that's a user option. Personally, I have not accepted the default, and, in some cases, I have employed the VCard when creating a new record.

The last option contained in this frame is to ☐ **Mark incoming messages as private** which is not selected in the default configuration. Selecting this option would mean that only those users that possess Master Rights, and this user would have the ability to read the text of an incoming Internet E-mail Message to this user. Usually selecting this option would be contrary to the usage of GoldMine as it was intended. All users that must deal with the contact should have access to all of the information that was generated with regard to the contact. However, there are times that privacy is a must. I would suggest that the user not select this option, and that, in those few cases where privacy must be maintained, the user take the time to mark the individual message as Private. There is an icon in the **Read E-mail** dialog that will make the message **Private**. Just click on the **Make Private** icon in the **Read E-mail** dialog.

This, then, concludes the **Retrieval Options** frame, and could result in the following possible settings being stored in the UserID.ini:

```
[Internet]
ReadOnGet=0
UseHeaderDate=0
UUEncodeScan=1
LinkOnGet=0
PreviewLines=100
KeepUserVCard=1
MarkIncomingAsPrivate=1
```

Now we can move right on to the frame that covers **Background E-mail Retrieval**. The user will notice that all of the options, save the first option, are greyed out (disabled) in the default state. All of the other options are dependant upon the user selecting the ☐ **Retrieve mail every xx minutes**. In fact, if the user set up to **Auto-retrieve**, under the **Accounts** tab which I discussed previously, then GoldMine has already predetermined the Auto-retrieve interval.

In the UserID.ini, the user should already be able to see an entry like:

```
[Internet]
GetInterval=15
```

Recommendation

I never recommend an interval of less than 5 minutes. No matter how tempting the urge, do not reduce the interval to 1 minute or you will end up with retrievals colliding into each other eventually.

Therefore, if the user has already selected to Auto-retrieve, then the user would be wise to define the message retrieval interval by selecting this option. Once this option is selected, all of the other options in this frame will become enabled for the users choosing. The user should be aware that the lower the interval number the more the network traffic will be increased. Should the network traffic prove to be an issue, then the user may be requested to increase the interval by their Network Administrator or their GoldMine Administrator.

Note

*GoldMine does not mark messages on the Server as having been Read when they are downloaded. Even checking this option, if the user has not selected **Auto-delete**, then they will download the same message over and over and over again until they do eventually delete the message from the server.*

The next option that should be enabled now is to ☑ **Skip read mail (recommended)** which, as the user can clearly see, is selected by default, and is (recommended) by GoldMine. Accepting the default setting would tell GoldMine, should there be any read e-mail on the Internet Mail Server, that GoldMine should ignore those messages, and to not download them again. However, if the user had set the option, under the **Accounts** tab, to **Auto-delete**, then this would be a moot point as all e-mail that was downloaded from the server should be deleted from the server, and should not appear on the mail server as a read message (refer to sidebar Note). It is redundant, but it has no side effect

that should be of any consequence. I would like to point out, again, that these settings are only in effect during the **Auto-retrieval** of e-mail messages. The user will still be allowed to download the read messages from the server by request, if they maintain this option setting.

The users next option/setting combination is ☐ **Skip mail larger than XXXX KB**, and the user can see that the default size is set to 1024 KB. Frequently, the user will be sent large e-mail messages. To Auto-retrieve all of those messages could quickly fill up a hard drive as well as take an inordinately large amount of time. With this option select, the user will have the ability to control which of those large messages should be downloaded, and which should just be deleted. With the GoldMine capability to Preview Internet e-mail messages, while they reside on the server, the user could easily cull the pertinent messages from the advertisements.

Now the user has the ability to ☐ **Skip messages from contacts not on file**. During the Auto-retrieve process, the user may not want to receive messages automatically from unidentified contacts. GoldMine stores all Internet e-mail addresses in the ContSupp table, and GoldMine will search through that table prior to downloading a message to verify that the E-mail Address does, in fact, exist should the user have selected this option. This would leave all messages from unknown contacts on the **Internet Mail Server** until the user has time to determine, through the GoldMine Preview option, that the message is a pertinent message, and that it should, in fact, be downloaded.

The last option in this frame is to ☐ **Send queued messages**. During the creation of an Internet E-mail message, the user has the option to queue the message for later distribution. Should the user select this option, during the **Auto-retrieve** process, GoldMine will attempt to deliver any messages that are currently queued for sending.

All of these options in this frame are only relevant during the Auto-retrieval process, and allow the user more control over their Internet E-mail. With the use of these options, GoldMine allows some automatic options tempered by some manual options. It is the user, in conjunction with their GoldMine Administrator, that must determine how much is to be automatic, and how much is to be manual. The UserID.ini settings that accompany the various settings in this frame are:

```
[Internet]
ActiveAutoGetMail=1
GetInterval=15
MaxEmailSize=1024
SkipLarge=1
SkipNoAddress=1
SendQueueWhenAutoGet=1
```

Note

These options may best be defined on the Corporate level. As a GoldMine Administrator, you may wish to employ a User Override (discussed later in this chapter) as opposed to allowing the users to set these Options willy nilly. This may also position you better for doing your Corporate Backups of GoldMine.

This hierarchal structure can make searching for attachments, outside of the GoldMine environment, much easier.

There is one more setting, and a few more options on the **Retrieval** tab that I must cover. The first is the path for any **Internet E-mail Attachments**. By default, in the old days, GoldMine set the **Attachments directory:** to be under the GoldMine folder. Now a days, GoldMine is trying to follow the Microsoft Windows Standards, and, as such, you can see in figure 3-33, the default path for this user is **C:\ProgramData\GoldMine\MailBox\Attach**. Should the user, the GoldMine Administrator, or the Network Administrator, designate a different location for which you, or the user, should store their e-mail attachments, then this is the field for instructing GoldMine as to that location. In a network environment, many Network Administrators will want these in a subset folder under the network installation of GoldMine for easy backup solutions.

I will now cover the rest of the options as shown in Figure 3-33. These are all checkbox options that allow for the structured saving of attachments as opposed to having all of the attachments saved under a single folder. The first of these options is to ☐ **Organize the attachments in a folder by receiving user**. Choosing this option would cause GoldMine to create a subfolder under the previously defined **Attachments directory:** folder based on the UserID, in my case, DJ. Therefore the default path that GoldMine will use for my attachments, as I have selected this option, would be:

```
C:\ProgramData\GoldMine\MailBox\Attach\DJ
```

Further, the user may opt to have subfolders under their subfolder. Yes, it gets deeper with ☐ **Organize the attachments in a folder by received date year**, and even deeper yet as once this is selected another option becomes enabled. ☐ **Further organize attachments in a folder by received date month**. You may choose both of these options without having chosen the option discussed in the previous paragraph. One could end up with quite the structure attachment hierarchy. Using my example again with these options selected, the path for todays attachments would be:

```
C:\ProgramData\GoldMine\MailBox\Attach\DJ\2009\05
```

The next, and final option tells GoldMine to ☑ **Save attachments as linked documents**, and this option is selected in the default configuration. GoldMine maintains a **Links** tab with linked documents, spreadsheets, attachments etcetera for each contact record. It is a one-to-many relationship created in the ContSupp table, such that each contact record can have many linked records. In the

default configuration, GoldMine will strip the attachment out of an E-mail message, place it in the designated directory, and link it to the contact record associated with the E-mail Address. Future viewing of the attachment can be accomplished from the **Links** tab on the contact record. In effect, GoldMine becomes your **Document Management Center**.

These options might be represented in the UserID.ini as:

```
[Internet]
AttachDir=C:\ProgramData\GoldMine\Mailbox\Attach\
NewFilingMode=1
AddYearToAttachDir=1
AddMonthToAttachDir=1
LinkAttachToCont=1
```

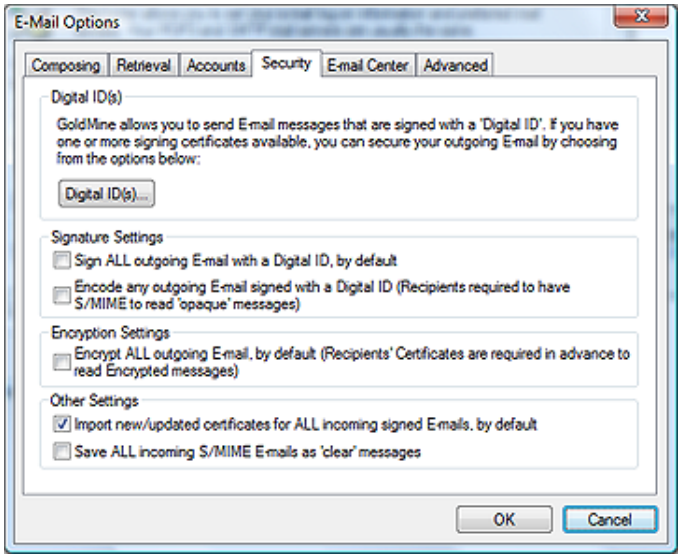


Figure 3-34

I have previously covered the **Accounts** tab, and I will, therefore, head directly to the settings and options covered under the **Security** tab as shown here in Figure 3-34.

This tab was newly added back in GoldMine 6.50.31113 and, as it states on the dialog form, *"GoldMine allows you to send E-mail messages that are signed with a 'Digital ID'. If you have one or more signing certificates available, you can secure your outgoing E-mail by choosing from the options below:"* If you own any Digital ID(s), and you desire to use them in your GoldMine application, follow the steps in the sidebar to prepare the ID for Import into GoldMine.

If you have imported a Digital ID into GoldMine in the first grouping, **Digital ID(s)**, you may want to set some of the other options that are available to you. The next frame of which has two options in it, is the **Signature Settings** frame. Here one may choose to ☐ **Sign ALL outgoing E-mail with a Digital ID, by default**, which does not, by the way, digitally sign an E-mail merge blast. Additionally, there is the option to ☐ **Encode any outgoing E-mail signed with a Digital ID (Recipients required to have S/MIME to read 'opaque' messages)**. This option should be self explanatory.

The next frame, **Encryption Settings**, has but a single option which is to ☐ **Encrypt ALL outgoing E-mail, by default (Recipients' Certificates are required in advance to read Encrypted messages)**. Again, I feel that the descriptor is very clear, and that no further explanation is required.

The last frame, **Other Settings**, allows the user to ☑ **Import new/updated certificates for ALL incoming signed E-mails, by default**, and is by default selected. With the usage of Digital IDs proliferating, one would probably want to maintain this setting in its default state. These are, however, user Options as always, so they are allowed to change these at will unless you impose a User Override (discussed later in this chapter). The last option is not selected in the default state, and allows the user to opt to have any encoded and digitally signed messages to be decoded and saved in a clear state. The ☐ **Save ALL incoming S/MIME E-mails as 'clear' messages** option allows the user to select this preference. One would probably want to select this option for received messages in GoldMine that are digitally encrypted.

With everything selected, the UserID.ini settings could look like:

```
[Internet]
DefaultSignSMIME=1
SMIMEUserOpaqueSign=1
DefaultEncryptSMIME=1
SMIMEAutoImpCert=1
SMIMESaveDecrypt=1
```

This allows us to proceed to the **E-mail Center** tab, Figure 3-35 on the next page. There are five separate frames contained on this tab, although three of the frames only consist of a single radio button option setting, as well there is a free floating option and an option button.

Steps

Prepare Digital ID for Import

1. *Open **Internet Options** via the **Control Panel***
2. *Click on the **Content** tab*
3. *Click on the **Certificates** button*
4. *Highlight the certificate that you wish to **Export***
5. *Click on the **Export** button, and follow the Wizard instructions, but making certain to save the export as a **.pfx** file*

Import Digital ID into GoldMine

1. *Click on the **Options** button from the Toolbar*
2. *Click on the **E-mail** tab*
3. *Click on the M**ore Options...** button*
4. *Click on the **Security** tab*
5. *Click on the **Digital ID(s)...** button*
6. *Click on the **Import** button*
7. *Find your **.pfx** file*
8. *Enter and confirm **Password:** if you have one established*
9. *Click on the **OK** button*

Note

*I need to make a statement here, I have no clue as to the value of **Digital ID(s)**. I did purchase one for testing with my GoldMine installation, and for the writing of this book, however, other than adding some extra steps to my e-mail process, I can not say that I have been able to see any added value to having, and using, this option. Quite the contrary, in fact, many recepients didn't know what to do with the digitally signed e-mails that I was sending to them.*

Recommendation

*I have found that the optimal solution is to not check ☐ **Show both E-mail Address and Account Name (if available) for Online accounts**, and to make sure that the users assign all accounts a colloquial account name that is meaningful to them.*

Note

*Quit franky, I would prefer the use of the ☑ **Show Outlook Folder** in the Mail taskbar option to that of using **GISMO**.*

Note

*Remember that I discussed, in a previous chapter, that the GoldMine Administrator may set the initial user Options for a network user, as they are setting up that user rights. The ☐ **Empty trash when closing E-mail Center** option is one option that the GoldMine Administrator may want to preset for the user, if they have not already set up a User Override to this.*

The first frame is the **General Settings** frame. There are two checkbox options within this frame, and both are selected in the default state. The first option, ☑ **Show both E-mail Address and Account Name (if available) for Online accounts**, may prove to be useful. Without this selected, GoldMine will insert the account name only in the **Online** tree, which is now located in the **Mail** TaskBar to the left in GoldMine Premium, if one is available, and if not available, GoldMine will insert the E-mail Address into the **Online** tree. With this option selected, both will be displayed.

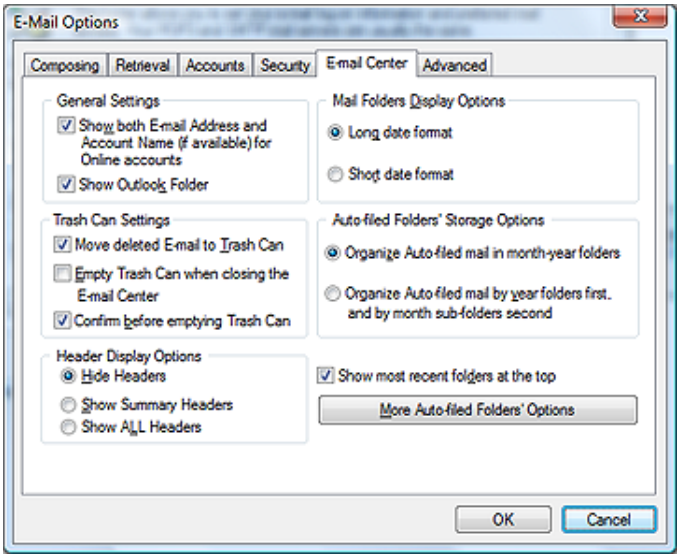


Figure 3-35

The next option in this frame is to ☑ **Show Outlook Folder** in the **Mail** TaskBar. I make no recommendation on this option. It can be a blessing as well as a curse. A bonus with GoldMine Premium is that on older versions of Outlook, one had to acknowledge that Outlook could be accessed for x amount of minutes each time Outlook was selected from the **Mail** TaskBar. With GoldMine Premium and Outlook 2007 this is no longer the case. Enabling this option will allow the user to Sync to/from Outlook, Copy to GoldMine, and Copy to Outlook by dragging from the one to the other, as well as being able to Zoom Outlook E-mail messages. If you are utilizing Outlook Business Contact Manager, these folders too will be displayed for the user in their **Mail** TaskBar.

Having said that, these two options could register in the UserID.ini as:

```
[Internet]
ShowFullAccountName=0
ShowOutlookInIMC=0
```

The next frame covers the **Trash Can Settings** options that are available for the user. The first option, of which, is to ☑ **Move deleted E-mail to Trash Can** which is selected by default. This presents a double safety for the user. When the user deletes e-mail, it is placed into the trash can. The user must then empty the trash can to completely remove the mail from their GoldMine. Even at that, GoldMine will present the user with the standard *"Are you sure"* message box before GoldMine allows for the emptying of the **Deleted** folder located in the **Mail** TaskBar. This option functions exactly the same, when selected, as the Trash Can which is located on the users Windows operating system desktop.

Caveat: You are in a SQL environment, and when a record is deleted in SQL, it is gone. You would most likely want to have your users employ this safety net option.

In conjunction with the last option, the user may also choose to ☐ **Empty Trash Can when closing the E-mail Center**. This option, when selected, causes GoldMine to prompt the user to empty the trash can as they close the **E-mail Center** if, and only if, the next option, ☑ **Confirm before emptying Trash Can** is also selected. Otherwise, when closing the E-mail Center, and this option is selected, GoldMine will just empty the trash can without a user warning. Yes this is cleaner, and more automatic, and recommended if you also select the next option. They may answer no to the question, and the trash can will not be emptied, but the important item here is that the question will have been asked. The user will have had to make a conscious decision to empty or to not empty the trash can. Once asked, the user must respond. A GoldMine Administrator may want their users, especially their users that do not concern themselves with standard file clean up procedures, to set this option forcing the thought upon the user. I would like to add that this is almost a moot option in GoldMine Premium as most people will leave the **E-mail Center** tab open if using the tabbed display, as will those that are using the Windowed arranged display.

The last option, in this frame, is selected by default, and forces GoldMine to ☑ **Confirm before emptying Trash Can**. This option goes hand-in-hand with the first two options, and is the redundant message that we had discussed. Should the user deselect this option, then they would be removing one of the safety layers that are available in the GoldMine **E-mail Center**. Accepting this option as is, will cause GoldMine to present a warning prompt, forcing the user to answer the **Yes** or **No** question when attempting to empty the trash can.

The setting and unsetting of these options could produce a resulting UserID.ini that might contain:

```
[Internet]
UseTrashCan=1
EmptyTrashOnExit=1
ConfirmEmptyTrash=1
```

If, in the UserID.ini, there was a setting of **UseTrashCan = 0** then the other two settings might not be displayed in the UserID.ini. Additionally again, for default values, GoldMine accepts the absence of a statement in the UserID.ini to represent the default setting, however, if someone were to enter by hand, **UseTrashCan = 1**, into the UserID.ini, then GoldMine would accept that value properly.

Note

*You may always choose to ⦿ **Hide Headers,** and the User will always have the option to Right-Click over the **Preview** area and select one of the other Header display options on the fly.*

In the next frame, **Header Display Options**, there is but one radio button option choice available. The user may choose to ⦿ **Hide Headers** information of an e-mail message, or the user may choose to just ○ **Show Summary Headers** information. With the last choice being to ○ **Show ALL Headers**. As these are a radio button selection option, the user may only select one of the three choices even though it visually appears to be two individual groupings. The summary information may include information such as the date, the sender, the subject as well as other information, but does not show any routing information that may be contained in the header. Usually all of the needed information is displayed in the **Online** folder of the **Mail** TaskBar, and in the **Preview** window. To include the summary, or all header information in the Preview window, could be considered redundant.

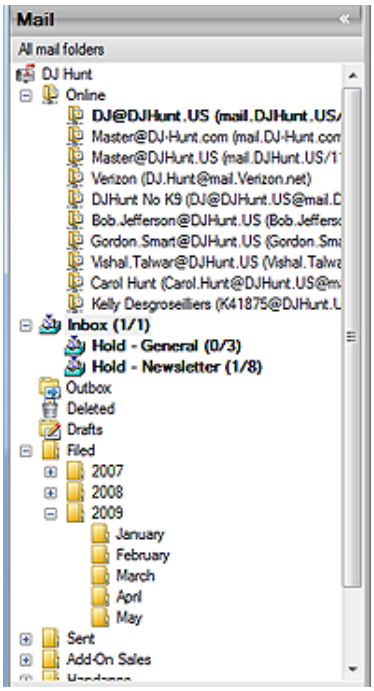
If the user had selected to show only summary header information, then the UserID.ini information could be:

```
[Internet]
ShowHeaders=1
```

In the next frame, we can see that, here as well, there is but one option, and again, it is a radio button option. This frame is the **Mail Folder Display Options** frame, and it is the option in this frame that allows the user to set the format for displaying the dates in the E-mail Center. Accepting the default ⦿ **Long date format** will cause the dates to be displayed as Wednesday, July 4, 2007 5:01 PM, while selecting ○ **Short date format** will cause the same date to be displayed as 7/4/2007 5:01 PM.

The result of selecting to use the short date format should appear in the UserID.ini as:

```
[Internet]
UseShortDate=1
```



Which brings up the last frame on the E-mail Center tab, the **Auto-filed Folders' Storage Options** frame. In this frame there is, again, but one selection from the two offered. The default selection is ⦿ **Organize Auto-filed mail in month-year folders** while the other option is to ○ **Organize Auto-filed mail by year folders first, and by month sub-folders second** (this second option selection is displayed for you here in Figure 3-36 as this is the way that I set my Options).

Both options are pretty much self explanatory. However, outside of this frame is a button that may have some relevance as to your choice here. The button is **More Auto-filed Folders' Options**, however, I'm getting ahead of myself. For some unknown reason, there is an option that affects the **Auto-filed Folders' Storage Options**, yet it is not included within this frame. This new option to GoldMine Premium 8.5.1 is to ☑ **Show most recent folders at the top**. Some users may wish to arrange their folders in the manner. Personally, I do not, hence, my folders are as shown in Figure 3-36.

Now let's return to that button. One would think that the button belonged within the afore mentioned frame as well, wouldn't they? And again, for that same unexplained logic, it is not. In Figure 3-37, on the next page, you can see the resulting dialog form produced by the clicking of this button.

Figure 3-36

One may choose to override the default month names for folders. To do so, they must be on this dialog form, and they must first select ☐ **Override the default month names** to enable the name fields for the user to be able to enter the overriding names.

These are the steps that I used to test these settings:

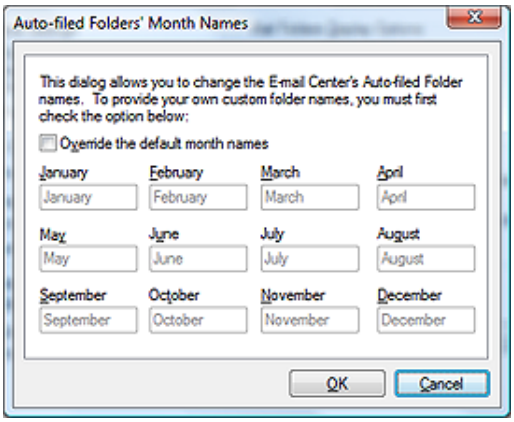


Figure 3-37

Step 1: I first left the default setting, and fast filed an e-mail.

Step 2: I then set the option to use year, and sub-folder month.

Step 3: I closed GoldMine.

Step 4: I restarted GoldMine, and then I fast filed another e-mail.

Step 5: I then overrode the month settings, and changed the month of December to Winter.

Step 6: I then closed GoldMine.

Step 7: I restarted GoldMine, and then I fast filed another e-mail.

Tip

*Should the tree in the **Mail** taskbar, Figure 3-36, get a different set of folders (new) when the user changes their **Options**, they may move their messages from one folder to another until the unwanted folder is empty. At that point the user may simply right-click on the unwanted folder, and select **Delete Folder** from the local popup menu. GoldMine will not automatically transfer the messages from one style folder setup to a different style folder setup.*

Note

*The line wrap here in the **[Internet]** section of my ini file is only for presentation purposes in this book, and is actually one continuous line of code.*

Why did I do all of this? I did this to see the consequences of these actions upon the **Mail** TaskBar. Yes, the user can make these adjustments, but all previous settings are maintained in the **Mail** TaskBar forever, unless the user does some finagling in the **Mail** TaskBar itself. All of the messages were fast filed on the same day. In the tree under the **Filed** folder, was the folder, December 2009 which was the result of maintaining the default setting. Under the Filed folder on the tree again, I saw a folder 2009 with a sub-folder of December. This was the result of my having selected the second option. Finally, maintaining the second option, but overriding the December name, I saw yet another sub-folder under the 2009 folder called Winter. That's a mess of folders for the same period.

As the messages were saved in the folders as they were named at the time the message was Filed, all of the folders and their E-mails are maintained in the tree forever unless the tree and its contents are manipulated by hand (see **Tip** in sidebar).

Leaving the options as last set in my test, should add the following lines to a UserID.ini:

```
[Internet]
NewFilingMode=1
MonthlyFolderNames=January*February*March*April*May*June*
                   July*August*September*October*November*Winter*???*
```

The last tab of the **E-Mail Options**, is the **Advanced** tab, and is shown here in Figure 3-38. This tab is becoming more and more crowded with each new release of GoldMine. I suspect, in future releases, that FrontRange will need to change their approach and/or add additional tabs. The reader will notice that there are four separate option frames, and one orphaned setting at the bottom of the tab.

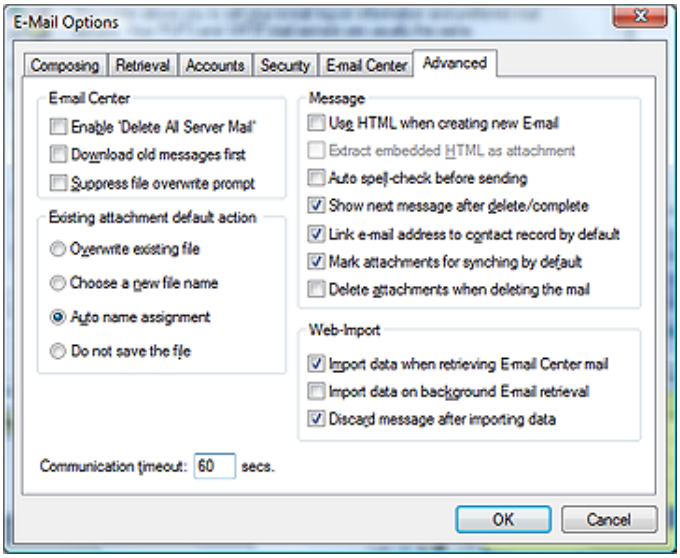


Figure 3-38

We may as well dig right in. In the first frame, we, again, have options that are pertinent to the **E-mail Center**. I won't even speculate why these options are on this tab as opposed to the **E-mail Center** tab, but they are here just the same, hence, we must deal with them here. In the default configuration, none of the options in this frame are selected. It is my opinion these settings should be controlled via a User Override.

WARNING

If, as an Administrator, you are using the server as a backup for e-mails, letting the user delete all server mail could be dangerous. As this is a user defined option, you will need to set policy in this matter, and regularly check that they are adhering to the policy. I would suggest a User Override.

Tip

The users will not know which attachment belongs to whom. It is, therefore, important that they have selected ☑ ***Save attachments as linked document****, see Figure 3-33, and that they use GoldMine as their* ***Document Management Center****.*

However, users or corporations may now choose to have each users attachments filed under separate folders. Hence this tip, although still valid, loses some of its significance.

Note

I only mention this here as an example. In fact, in my UserID.ini I have opted to ◉ ***Overwrite existing file****.*

I do this because I have a system (don't we all). Whenever an attachment is received, I store a copy of it in a client folder that I create for all of my clients. It is, therefore, only important to have a copy of the last received Lookup.ini, for example, in my ***Attach*** *folder under GoldMine.*

Note

The user must have the latest version of Microsoft Internet Explorer on their system. The GoldMine product is married to Microsoft Internet Explorer for many of its components. One major component is fonts in GoldMine are dictated by the fonts in Intenet Explorer. There is no alternative option for something such as FireFox for instance.

Your first option, in the **E-mail Center** frame, is to ☐ **Enable 'Delete All Server Mail'**. If the user selects this option, they are causing GoldMine to show an additional list in the local menu listing. This item will permit the user to delete all of the E-mail on the Internet Mail Server with the click of one button. Naturally, because of the potential danger, GoldMine will present the user with an **Are you sure** type of message.

Selecting the next option in this frame will instruct GoldMine to ☐ **Download old messages first** from the Internet Mail Server. Selecting this option really has no noticeable effect to the user, as the user E-mail is always sorted Newest to Oldest on the **E-mail Center** dialog form, at least as of this writing. There has been some suggestions put forth to have this as a user definable option.

Lastly, in the **E-mail Center** frame, the user has the ability to ☐ **Suppress file overwrite prompt**. In the next frame, the E**xisting attachment default action** frame, I will be discussing the options that the user is offered when the prompt is displayed. The user may choose here to not display the prompt at all, and GoldMine will not warn the user about duplicate attachment file names. GoldMine will simply follow the instructions set by the user in the **Existing attachment default action** frame.

The **E-mail Center** frame options under the **Advanced** tab could result in settings such as these in the UserID.ini:

```
[Internet]
AllowDeleteAll=1
GetOldToNew=1
SkipOverWriteUI=1
```

This next frame, the **Existing attachment default action** frame, has but one selection option as this is again a radio button selection. The user may choose to have the user overwrite prompt default to one of four possible choices. The user may choose to default the prompt to ○ **Overwrite existing file**, or to default to ○ **Choose a new file name**. The user may choose to accept the default setting for this frame which is to ◉ **Auto name assignment**, or, lastly, the user may choose to default the prompt to ○ **Do not save the file**.

Many times, users will receive attachments having the same name as one that already exists. In my case, I received hundreds of Lookup.ini files to review. Obviously, I hope, each one is important. I would not want to default my option to overwrite the existing file. I might opt for the second choice of choosing a new file name, however, that requires additional work. It would be much simpler, just to accept the default, and to let GoldMine do the ◉ **Auto name assignment**. With this setting, GoldMine will append the next consecutive number to a file name until it is unique or the number has reached 64 at which point GoldMine will attempt to generate a new name entirely for this particular file. Some of my Lookup.ini file names could look like Lookup(1).ini, Lookup(2).ini, Lookup(3).ini, etcetera. The user will be able to see these names in the folder that they chose for their attachments, which, as you'll remember, in the default configuration, was ...\GoldMine\MailBox\Attach.

Because this UserID.ini setting does not follow any standard convention, I will show you all of the possible permutations, however, only one entry is possible:

	[Internet]
○ **Overwrite existing file**	**RetrieveOverwrite=6**
○ **Choose a new file name**	**RetrieveOverwrite=7**
◉ **Auto name assignment**	**RetrieveOverwrite=4**
○ **Do not save the file**	**RetrieveOverwrite=2**

Let's move on to that next frame then. This frame is the **Message** frame, and permits the user some options directly affecting their messages. The first option in this section, for instance, tells GoldMine to ☐ **Use HTML when creating new E-mail**, which, surprisingly, is not checked in the default state. Well, as far as I can tell, although this is an option, in GoldMine Premium selecting or deselecting this option has little effect. GoldMine Premium honors the setting of the E-mail Address **Rich text (HTML)**, and, if an HTML e-mail is received, GoldMine will receive it in HTML format. This is a big change over previous versions of GoldMine. FrontRange is constantly revamping, and enhancing its e-mail capabilities. Quite frankly, in todays Internet, I would always, and do always select this option.

This next option, ☐ **Extract embedded HTML as attachment**, is not even available for selection. It is disabled. Again, this is a legacy setting that they just didn't bother removing from the GUI.

The next option is really self explanatory, however, this book requires that I cover each item. ☐ **Auto spell-check before sending**, if selected, will automatically check the body of the e-mail message for spelling errors using the built in GoldMine dictionary as it is preparing to send the message. My spelling is horrendous so I always have this option selected.

The ☑ **Show next message after delete/complete** option is a power user option, and is selected in the default configuration state. Once the user has finished with the message they are reading, and

delete or complete it, GoldMine will automatically present, in the e-mail reader, the next sequentially listed message contained in the **E-mail Center Inbox** folder. Personally, I do not like this feature, and I have deselected this option for myself. You have to love that these are User Options.

☑ **Link e-mail address to contact record by default** is also selected in the default configuration state. If selected, GoldMine will attempt to link the E-mail Address, from the retrieved message, to the contact record to which the message has been linked. If the user wishes to link a message to a contact record, and to not have the accompanying E-mail Address linked as well, then they should remove the check from this option. I can't think of when or why a user would not want to link an E-mail Address to a contact record, however, this is a user option for messages. I can only assume that there were enough requests from end users for FrontRange to have included this as an option within the GoldMine user Options.

☑ **Mark attachments for syncing by default** is another user Option that is selected in the default configuration state. In a synchronization environment this might be a good option for selection. With this option selected, all linked incoming (this is an important distinction for you to remember) E-mail attachments will be marked for synchronization. The GoldMine Help file states, *"...except for VCards and transfer sets."*, however, these items are not usually linked to a contact record anyway. If, after receiving the message, the user decides against synchronizing the attachment, the user may go into the attachment, under the Links tab of the contact record, and deselect the option ☑ **Allow File to Synchronize**. If the user, or organization of users, receives an abundance of attachments, and they are all selected for synchronization, this could cause an extremely long synchronization time. This would be much more noticeable if the user happens to be synchronizing via a modem connection (Does anyone still use dial-up?). I can guarantee that, in this situation, it won't take the user long to decide to remove the check from this option.

And the final option for the **Message** frame is to ☐ **Delete attachments when deleting the mail**. This is not selected in the default configuration state of GoldMine. I have seen, too often, many orphaned attachments when this option is not selected. Hard drives, even with the volumes they can handle today, can easily and unknowingly get completely filled up with attachments. If the user is deleting the E-mail message, then they are probably also through with the attachments as well. The users should decide, together with their Network and/or their GoldMine Administrator, whether this option should be selected or not. I have this option selected for my team. I have trained my users to save a copy of any attachment, through the E-mail reader, that they need to have saved, when they plan on deleting the E-mail message.

Well, that was a rather large option frame, seven items in all. Albeit, two that were irrelevant. I would like to reiterate to make some of these UserID.ini settings available for display here, I had to reverse their default setting. Again, I emphasize, not all settings are simply a 0 or a 1. Some settings are not contained in the UserID.ini unless they are in their non default state.

Here are the various settings that might appear in the UserID.ini for the **Message** frame:

```
[Internet]
UseHTMLByDefault=1
ExtractEmbeddedHTML=1 ( No longer entered thru GUI )
AutoSpell=1
SkipOnDispose=0
DefaultLinkAddr=0
SyncAttachmentDefault=0
DelAttachWithMsg=1
```

Our fourth and final frame on the **Advanced** tab is the **Web-Import** frame which consists of three user options. A WebImport file is a specially constructed E-mail message. This message will usually be constructed by a perl, php, or asp script on a website, and the E-mail is then sent to GoldMine with special header information: **{$GM-WEBIMPORT$}**. This special **To:** information identifies this message as a WebImport message which is to be handled differently by the GoldMine E-mail Center. In this frame the user is instructing GoldMine as to what should happen with regards to any WebImport message. In the default configuration, the user option is set to ☑ **Import data when retrieving E-mail Center mail**. This causes GoldMine to import data at the same time as it is retrieving any E-mail messages from the Internet Mail Server. In an ideal environment, the GoldMine Administrator would not have WebImport E-mail directed to an individual user, but instead, would direct this E-mail to someone who is tasked with this job function. The GoldMine Administrator may ask most users to turn this option off.

The next option in this frame pertains to unattended E-mail retrieval. Should GoldMine ☐ **Import data on background E-mail retrieval**? Background e-mail retrieval is better known to users as Auto-Retrieval, and, as you can see, this option is not selected. If this user were the designated user for WebImporting, then the user may wish to have this option selected, so that anytime their Internet mail server is checked for E-mail, the WebImport E-mail will be processed as well.

WARNING

Users Do Not Read Warnings

*With this option selected, if the E-mail Address already exists within GoldMine, GoldMine will tell the user that the E-mail Address is linked to another record. GoldMine will ask if they want to move the E-mail Address to the current record. Trust me. The user will just click **OK** without reading the message. You will end up with many mislinked E-mail Addresses in your GoldMine database. You really, really, really want to consider employing a User Override here to make certain that the ☑ **Link e-mail address to contact record by default** option is **not** selected.*

Note

*With todays version of WebImport, you may have the marker, **{$GM-WEBIMPORT$}**, anywhere in the E-mail message. This is because you must set a **Rule** in your GoldMine that will designate that any e-mail with this marker as a WebImport message, and should be processed accordingly.*

Note

Background E-mail retrieval may also be induced through the commandline use of the Server Agent. This silent mode usage would be set into motion with the use of a commandline similar to:

. . . \ G o l d M i n e \ G M W . e x e /u:UserName /p:Password /s:DDE

Note

*Having said that, you should be aware that the WebImport instruction set could include a **Save** statement which is independant of this switch. In fact, I save all WebImports to the ContHist table as a backup to the updated/new information incorporated as a WebImport.*

The last option in the **Web-Import** frame, ☑ **Discard message after importing data**, is also selected in the default configuration state. There is not really any need to keep the message once a record has been created from the e-mail message, except during the initial testing phase of your WebImport process. The information contained in the message becomes redundant information (see sidebar Note), and clutters one of GoldMines fastest growing tables, the **Mailbox** table. The user should discuss any desire to change this option with their GoldMine Administrator prior to making the change.

These three options could show up in the UserID.ini as:

```
[Internet]
AutoWebImport=0
BackgroundWebImp=1
DiscardWebImportMessages=0
```

Not in any frame, but still on the **Advanced** tab, is the setting for **Communication timeout: xxx secs.**, where in the default configuration, the xx is actually **60** seconds. On forced retrieval or Auto-Retrieval, GoldMine will attempt to connect to the Internet Mail Server. If it can not connect within the allocated time, GoldMine will break off its attempt, and usually issue a Connection Failure error message. On slow connections, one might need to bump this setting up. I have seen that setting this to 120 has helped in some client situations. Personally, I have mine set to 1440 seconds.

This setting would appear in the UserID.ini as:

```
[Internet]
TCPTimeout=60
```

Wow, that was one long section, wasn't it? On all of these last few pages, I have only been discussing the **E-Mail** tab, and its many subtabs. I thought that this would be a long section, but I never figured it would turn out to be this long. Well, we have now concluded with the **E-Mail** tab, and all of its options. It is time to move on to the next tab, the **Telephony** tab.

Telephony

Note

*Modem settings, under this tab, are employed mainly when using GoldMine to dial the active Contact record. From the GoldMine menu that would be: **Action** | **Call Contact** ► | **Dial***

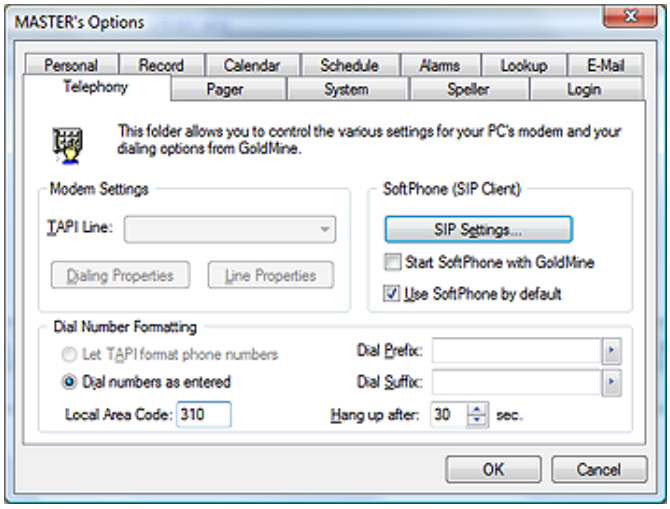


Figure 3-39

The tab, shown here in Figure 3-39, is the **Telephony** tab, and it is this tab that allows the user to control the choice of which PC modem to employ should the user have more than one set up. As you are aware, default modem and dialing properties are set up in the **Phone and Modem Options** under the **Windows Control Panel**. GoldMine also allows the user some additional control over the default settings that GoldMine will employ only in its use of the chosen modem.

The first of the three frames that I will discuss is the **Modem Settings** frame. The user is expected to select a **TAPI Line:** from the drop-down list of their systems pre setup modems. If the user only has one modem available on their system, this modem will already be selected in this field for this user.

I will not be discussing the **Dialing Properties** button or the **Line Properties** button in this book as both are pointers to the Windows options for the selected modem. The user should have no need to enter into either of these areas except for one, and only one reason. If you will look at Figure 3-39 in the **Dial Number Formatting** frame, you will see one field that contains information. This is the **Local Area Code:** setting. Unlike past versions of GoldMine, this information can now be modified directing within the GoldMine Telephony dialog form. This field is read directly from the dialing properties of within Windows. Any changes to **Local Area Code:** setting should be made through the users Windows environment, however, it can now be overridden via the UserID.ini. After any change has been made in the next frame, however, some of the options from this frame will be written to the UserID.ini.

```
[Modem]
DeviceID=0
AreaCode=978
```

Note

*In order for the TAPI option to work properly when using GoldMine to dial an international number, the GoldMine **Edit** | **Record Properties** ► | **Record-related Settings...**, **Phone Formatting**, must be set to ◉ **Non-USA Format** for that particular contact record. GoldMine.*

Note

*Windows uses the **Telephone Application Programming Interface** (**TAPI**) to set up modems.*

I should now finish off by discussing the rest of the **Dial Number Formatting** frame options. The first option is a radio button selection option. The user has the choice to ◉ **Let TAPI format phone numbers** or to ○ **Dial numbers as entered**. The default selection is to let TAPI format the phone numbers. Letting TAPI control the formatting of numbers means that, should the **Country:** field of the GoldMine contact record contain a Windows recognized country, then TAPI will automatically supply the appropriate access and country codes when GoldMine is dialing the number stored in the **Phone1:** field of the contact record. Let's say that the **Phone1:** field contains **03 95659780** and the **Country:** field contains **Australia** on a particular contact record. When dialing this number through GoldMine, and under the TAPI rules, the international access code of **011** would be supplied as well as the country code of **61**.

The alternative selection here is to ○ **Dial numbers as entered** which, believe it or not, lets GoldMine dial the number as entered. However, the selection of this option will also cause GoldMine to adhere to any of its own rules including prefix or suffix entry as well as any settings that may have been instituted in the **PreDial.ini**. I will not be discussing the **PreDial.ini** in this book, however, I would suggest that you, the GoldMine Administrator, look at: **Technical Document 387**, **Using and Setting up GoldMine's for Special Dialing Needs** at http://Support.FrontRange.com/Support/GoldMine/387_PredialINI.htm.

I've already discussed the fact that the user can not change the system **Local Area Code:** field from within this frame in GoldMine, so let's move on to the **Dial Prefix:** field. The prefix is a series of numbers and characters that must be dialed before dialing the actual telephone number. Some prefixes that come to mind are 9 to access an outside line, a comma (,) to enter a 2 second pause, or *70 to disable call waiting. Whatever string the users particular circumstances dictate, this is the area to enter the prefix information.

In that same venue, we have the **Dial Suffix:** field. Here the user would enter a number sequence that executes special commands, and must be dialed by the modem after the telephone number has been dialed. In some offices this might be the extension number of the telephone from which the number is being dialed. This occurs for accounting purposes, for example. Alternatively, the reversing of the previously mentioned *70 command, to disable call waiting, would be *71, to enable call waiting.

Lastly we have the setting which instructs GoldMine to **Hang up after: xx sec.** which is set in the default configuration to **30** seconds. Once GoldMine attempts to make a connection, if the connection is not made within this allocated time frame, GoldMine will hang up the modem, freeing the telephone line for other usage. FrontRange has determined that 30 seconds is the optimal time for most users, however, the user is free to raise or lower this number as they deem adequate for their particular usage.

The remainder of the possible UserID.ini settings for this section are shown here:

```
[Modem]
TAPITranslation=1
ModemPrefix=*70,, 9,,,
ModemSuffix=*71
HangupTime=30
```

Note

*I am only discussing the **SoftPhone** as it is part of the **Options**. As I do not have, nor use VoIP, I have not had the capabilities of testing this feature in GoldMine Premium.*

The **Telephony** tab used to bear the name of **Modem** as its tab name. This was changed back in GoldMine 6.7. In addition to this change, a new frame was added to the tab, the **SoftPhone (SIP Client)** frame. This frame allows you to use, and to configure a VoIP (Voice Over IP) system that you may have in place (in theory). I don't have this feature enabled on my telephone system, hence, I am not able to test this out for you. I can only show you the ini settings derived by making changes in this frame.

The first option in this frame, see Figure 3-39, is the **SIP Settings** button. Clicking on this button brings up the dialog form shown here in Figure 3-40. I have taken the liberty of filling in this form with information so that I could show you the result stored in the UserID.ini. These values are meaningless, and are only here as reference markers.

You'll also notice, in Figure 3-40, that I have entered a Password, and, appropriately it is hidden

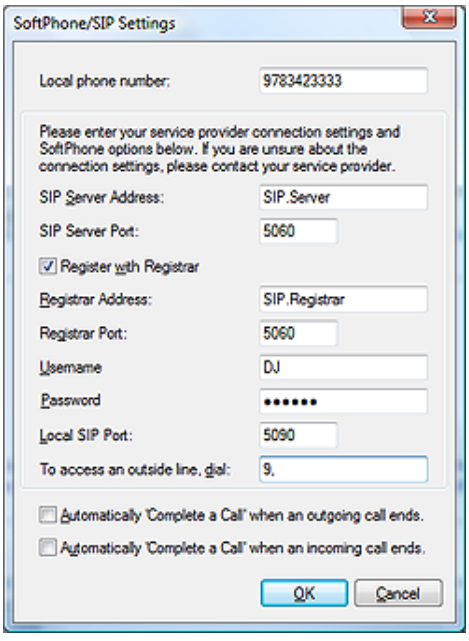


Figure 3-40

in the dialog form. However, the same does not hold true for the UserID.ini. Review the **[SoftPhone]** section of the UserID.ini displayed below, and you will notice the statement:

```
Password=waldo
```

This looks suspiciously like clear text to me. Whoops! We have a Security breach.

Back to Figure 3-39, you notice two more options that could be utilized if you are using VoIP with your GoldMine, the first option is to ☐ **Start SoftPhone with GoldMine**, while the other is to ☑ **Use SoftPhone by default**. I would believe that both of these options are self explanatory.

Not that these values represent anything that is accurate, but to show how these settings are reflected in the UserID.ini:

```
[SoftPhone]
Extension=9783423333
PrimaryRegistrarName=SIP.Registrar
PrimaryProxyName=SIP.Server
ContactDialingPrefix=9,
Username=DJ
Password=waldo
RecordingdDir=
VoiceOutDevice=
SysTonesDevice=
VoiceInDevice=
ProxyDomain=
PrimaryRegistrarPort=5060
PrimaryProxyPort=5060
SipPort=5090
TypeOfService=0
RecordingBuffer=0
Register=1
CreateHistOut=1
CreateHistIn=1
BypassNAT=0
AutoBind=1
```

Pager

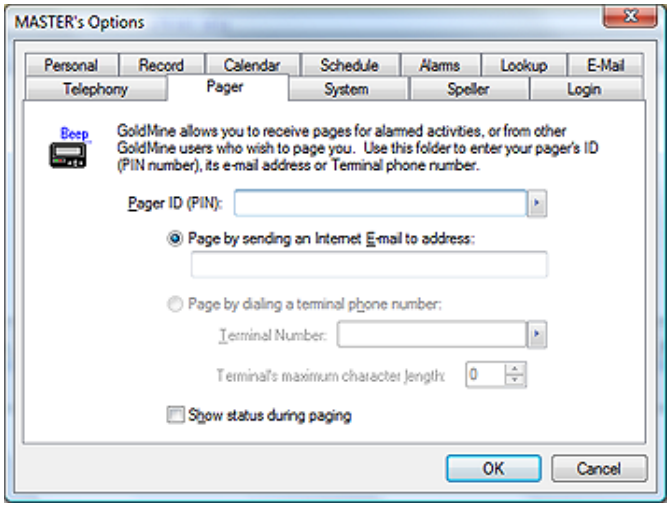


Figure 3-41

Well, that was rather a short, and relatively painless tab to cover. The next tab, the **Pager** tab, is even shorter. Pager? Does anyone even use one of those any more? As one can see in Figure 3-41, there are only really two items that the user needs to concern themselves about. The reader should remember, when I covered the **Alarms** tab, Figure 3-23, there was an option to ☐ **Page me with the alarm when not acknowledged within: 10 minutes**. If the user has selected this option, or if the user wishes to allow another GoldMine user to page them, then they must complete the information contained on the **Pager** tab.

The first item is the **Pager ID (PIN):** field into which the user should enter their pagers PIN. Should the user not know this information, they may need to contact their Pager Service Provider for the required PIN information.

Once the pager ID information has been entered, the user has to select one of two distribution methods. The first method, which, in GoldMine Premium, is the default selection, is ◉ **Page by sending an Internet E-mail to address:**. If this is the users choice, then they must enter the Internet E-mail Address to which they wish to have the page sent. In GoldMine Premium, the second option appears to always be disabled (see Sidebar Note).

On the other hand, if available, one may select ○ **Page by dialing a terminal phone number:**. When the user selects to dial a terminal phone number, they must include the **Terminal Number:**, and then there is one piece of additional information the user must supply for GoldMine, the **Ter-**

Note

Although the second option is disabled via the GUI, I believe that you may still enter the statements manually via the UserID.ini.

In order to enable this option, you may have to have a Modem connected and configured in your operating system. I certainly don't so I can't say for certain.

minal's maximum character length: xx which is set to 0 in the default configuration. If the user doesn't change this number, then the user will not receive any messages. Most pagers will accept 80 to 100 characters of information. The user should enter the greatest number of characters that their pager can accept at one time.

There is one more option that we must talk about in this build of GoldMine, and that is whether to ☐ **Show status during paging** or not. Naturally, if selected, this would be displayed in the Process Monitor within GoldMine.

There is only one possible UserID.ini set of settings that could be generated based on the information contained on the **Pager** tab. This set might look like:

```
[GoldPager]
PIN=012345678
PagerEmail=DJ@DJHunt.US
MaxChars=0
ShowStatus=1
```

Notice that the MaxChar statement is added to the UserID.ini even though it will not be employed with this setting. While the manually entered **Terminal Number:** might look like:

```
[GoldPager]
PIN=012345678
Terminal=9,,19783423434,,,6
MaxChars=100
ShowStatus=1
```

I would mention here, in the older versions of GoldMine, the next tab would have been the **Sync** tab. In point of fact, I had originally covered the **Sync** tab here in our first draft of this chapter. As FrontRange has chosen to remove these UserID.ini setting from this GUI in GoldMine 6 and later, I too have decided to remove them from this chapter. They are, however, covered in the **GoldSync** chapter later in this book.

System

We're almost to the end of the user Options' tabs. Only three more left to cover if the user has Master Rights, and only two more for those users who do not possess Master Rights. The next tab that I will cover, Figure 3-42, is the **System** tab (prior to GoldMine Premium, this was known as the **Misc** tab), and there are four frames on this tab, the **Navigation** frame, the **Date and Time** frame, the **Classic toolbar** frame and the **Help** frame. There is also one free floating option that I will discuss shortly.

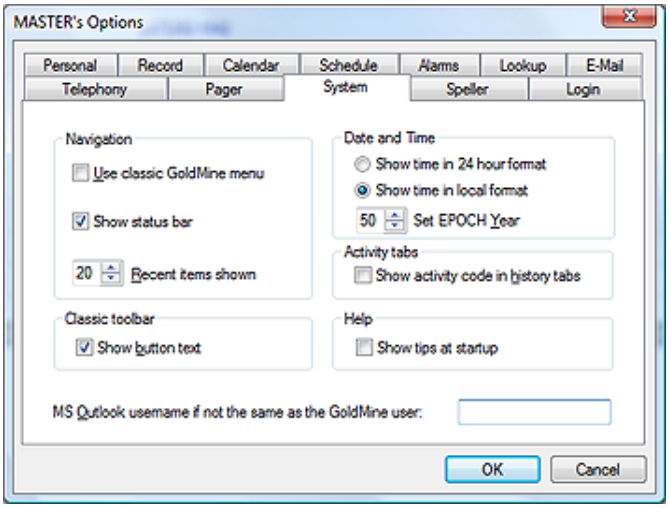


Figure 3-42

Let's first begin with the **Navigation** frame which contains two checkbox control options, and one spinner control option. The first of these is to ☐ **Use classic GoldMine menu**, and I do not recommend that you select this option. Yes your users will have a learning curve to overcome with the new, and supposedly more "User Friendly" interface, but if you don't maintain the status quo of GoldMine Premium then your users will never conquer that learning curve. Alas, they will remain in the GoldMine of their youth. I strongly recommend that you, as the GoldMine Administrator, utilize a User Override which I will discuss later in this chapter.

The second option is to ☑ **Show status bar**, and it works in GoldMine Premium 8.5, sort of. Okay, in previous versions of GoldMine the option was called ☑ **Show Taskbar,** and didn't function. In this version of GoldMine Premium, if this option is selected, then you may change the setting as you wish, and the changes will be apparent immediately. On the other hand, if you are in GoldMine without this option selected, and you decide to change the option to show the status bar, then you will have to exit GoldMine and re-enter GoldMine before the screen refresh will occur. All in all, why would anyone not want the information display in the status bar?

The first option in the **Date and Time** frame is to O **Show time in 24 hour format**, and I do believe that this is another of those self-explanatory options but... Time is displayed within GoldMine at var-

ious locations, one of which is the Status Bar at the bottom right of the main GoldMine screen. The 24 hour format is more commonly known as military time. In the default configuration, GoldMine will display all times as 9:00am or 9:00pm. These two times would translate into 09:00 and 21:00, respectively, in the 24 hour format (and represents how time is stored in the GoldMine tables as GoldMine has not yet made use of the SQL DateTime field types). Changing this option would, in no way, affect the way that time is stored in the tables. Its selection, or not, only affects the manner in which time is displayed within the GoldMine display environment.

The next option that the user may set is to ◉ **Show time in local format**. Do you remember when I discussed the **User-Defined Date** field formatting earlier under the Records tab? Choosing this option does not appear to override that option, and, in fact, when selected, I cannot find were this visually affects any of the dates within my GoldMine except possibly the Status Bar.

Note

Corporations in this century, that have purchased GoldMine for the first time, do not really need to concern themselves or their users with the EPOCH setting.

The next option is to **XX Set EPOCH Year**, and is a spinner control that increments and decrements in units of five at a time when using the spinner control. The user may just type a number into the field if they find that to be easier. This setting came into being to confront the **Y2K** issues that were expected around the turn of the century. A lot of users had upgraded their GoldMine through the years, and a lot of the older dates were stored with two digit year dates. This was the solution that was developed to instruct GoldMine in how to interpret those two digit year dates. The default setting is **50**. Whatever this setting is, GoldMine will consider any two digit year that is less than the number to be in the current century, and, conversely, any number that is larger than or equal to the number to be in the previous century. Therefore, with the default setting of **50** the date 6/17/49 would be reinterpreted to represent 6/17/2049 while 10/11/52 would be reinterpreted to be 10/11/1952. This will have no affect on years that are stored in the GoldMine tables with four digit years, and will only affect old time users of the GoldMine product who have been continuously upgrading. New GoldMine Premium installation users should probably just ignore this setting. Hmmmm! If anyone still has the old two date years at the end of the start of next century this could cause issues. Who cares, we won't be around to see it.

New to GoldMine Premium 8.5 is the addition of three single option frames. And the first new addition is the **Activity tab** frame with its' single option of ☐ **Show activity code in history tabs**. You long time readers of my books will remember this as a GUIless ini setting described in that section of this chapter. In fact, here is the old write up:

■ Tabs - Displaying the Activity Code under the History tab

I use Activity Codes extensively in my organization. I use them for invoicing, for projects, as well as Contracts. I would like my users, when looking on the **History** tab, to be able to see the activity code for the various activities. I do not want them to have to take the extra steps of zooming, and closing the zoom, just to read the Activity Code. To facilitate this, I inserted the following statement into each UserID.ini (actually I did a User Override in the GM.ini):

```
[CalObj]
HistShowActvCode=1
```

This is much more informative at a glance, and I find it to be very helpful in the way that I have defined my organization, and its usage of GoldMine Premium.

The next options that we must talk about is the **Classic toolbar** frame with its' one option of ☐ **Show button text**. Choosing this option will display any classic buttons, like when reading your E-mail messages, about 4 times the default size, and display up to two lines of text about each buttons capability. I do not recommend the selection of this option as it forces some buttons off screen. I prefer the default setting, and, if I am unsure of the buttons function, I simply hover the cursor over the button 2 milliseconds until the Tool Tip displays the buttons functionality. This particular option is contained in a different section of the UserID.ini, hence I display it here:

```
[Toolbar]
ButtonText=1
```

In the final frame, **Help**, our one option is to ☐ **Show tips at startup**, which is itself, self-explanatory, and when selected would appear in the UserID.ini as:

```
[Warning]
ShowTipOfTheDay=1
```

Now let's move on. And, lastly, on the **System** tab at least, we have a field for the **MS Outlook username if not the same as the GoldMine user:**. This is important, if for instance, the user desired to look at their Outlook information from within the E-mail Center, and the user maintained a different username than that which they have as their GoldMine username. In the default configuration, GoldMine will attempt to access the Outlook application using the GoldMine username. The user is encouraged to assign the same username for both their GoldMine and their Outlook. However, if

they do not have the same username on both systems, then this is the field in which the user is to tell GoldMine the username to employ when accessing their Outlook data.

The result of having set all of these options under the System tab could show in the UserID.ini as:

```
[GoldMine]
GMClassicMenu=1
ShowStatusbar=1
DateInLocalFormat=1
TimeIn24Hr=1
EPOCH=50
MSMailUser=Waldo
```

Speller

The next tab that one encounters is the **Speller** tab. FrontRange had switched their spell checking software to a more robust model back around GoldMine 6. This tab, though it contains user options, does not store these options in the UserID.ini that I have been discussing so far, the one that is in the root folder of GoldMine. These dictionary settings are also stored in a UserID.ini, but these are stored in a username folder under the **Speller** folder which is under the root GoldMine folder. So if the **UserID** of the user were **DJ**, they would have a file like **...\GoldMine\DJ.ini** as well as a file like **...\GoldMine\Speller\DJ\DJ.ini**. While you are examining the latter folder, you will also notice that there is another file here like **...\GoldMine\Speller\DJ\DJ.tlx**.

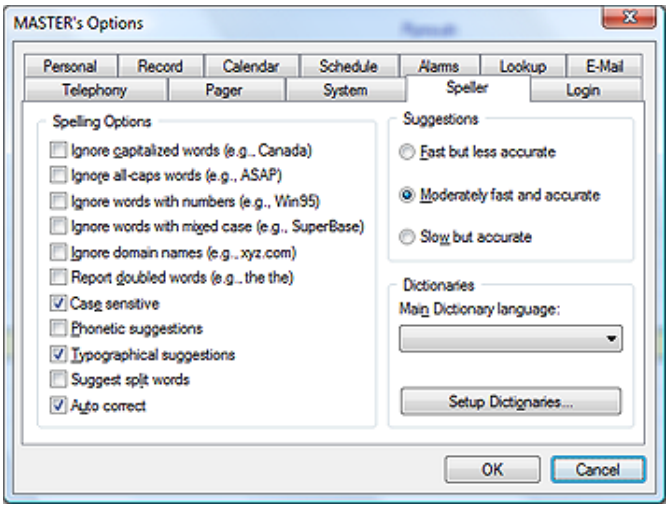


Figure 3-43

Note

*The ***.tlx** file contains the users personal dictionary, and may be edited in GoldMine Premium using the **Setup Dictionaries...***

Looking at Figure 3-43, one will ascertain that there are three different frames. The **Spelling Options** frame with its eleven checkbox options, the **Suggestions** frame with its one radio button selection, as well as the **Dictionaries** frame. Let's take a closer look at the first frame, **Spelling Options**.

The first checkbox option in this frame is to ☐ **Ignore captialized words (e.g., Canada)**. This clearly instructs the spell checker to ignore, what is known as, proper names.

The next option is to ☐ **Ignore all-caps words (e.g., ASAP)**. Again, with the given example, this is fairly clear. If selected, the spell checker will ignore acronyms such as the shown example. Also initialized names such as DJH will be ignored.

The user, in the next option, may choose to ☐ **Ignore words with numbers (e.g., Win95)**. Again, we have the obvious. The examples given with each option are very descriptive on this tab of the user options GUI. By the way, had I selected the previous option, the word GUI would have been ignored when spell checking this chapter. In this case, if selected, and the word contains numbers, the spell checker would ignore the word.

The next option comes into play quite a lot, especially where the name of the product is GoldMine. The user may select to ☐ **Ignore words with mixed case (e.g., SuperBase)**. Okay, you ask: "Why wouldn't you think that, as an example of mixed case, FrontRange would have used the word GoldMine or FrontRange?". To which I would have to answer: "I have no clue.". Many times, one will employ the camel back style word in their notes, and it may be beneficial to ignore these when performing a spell check.

Plodding forward, we see the option to ☐ **Ignore domain names (e.g., xyz.com)** which would always be interpreted as a spelling error. I always select this option as it will ignore domain names as well as Internet addresses.

As I make this error often, the next option is another that I would always select. It is the option to ☐ **Report doubled words (e.g., the the)**. Not **that that** is a problem for us, but I like to select it anyway.

The next option is selected in the default configuration. The ☑ **Case sensitive** option is selected. With this option selected, the spell checker will evaluate the word Maple as being different from the

word maple. If one or the other is not in the dictionary, then the one not found will be considered as a spelling error.

☐ **Phonetic suggestions**, if selected, would have the spell checker offer, in the suggestion list, words that have a similar soundex value. In laymans terms, words that sound the same as the targeted word.

☑ **Typographical suggestions** is selected, in the default configuration, and has the spell checker offer suggestions of words that have a similar character content as the targeted word. This helps to pick up spelling errors with transposed letters that often happens with flying fingers on the keyboard.

☐ **Suggest split words** is the next option, and, again, it is not selected in the default configuration. I'm going to just go with the GoldMine Help file definition here, as I have not quite deduced what this actually means. *"Suggest split words: Suggests two words for compound words that do not appear in the dictionary."* What I believe this to means is, if the user were to type in bigman, then the spell checker would suggest big man.

☑ Auto correct is an option that lets the spell checker automatically correct words if they are defined as *"Auto Change (Use case of other word)"* or *"Auto Change (Use case of checked word)"* when classifying the Action: for any words that one defines in their own dictionary. When not selected the user will be prompted before any word, so designated, is changed.

This ends my discussion of the **Spelling Options** frame. The UserID.ini settings that are affected by changing these settings are shown here:

```
[SSCE User]
IgnoreAllCapsWords=0
IgnoreCappedWords=0
IgnoreDomainNames=0
IgnoreMixedDigits=0
IgnoreMixedCase=0
IgnoreNonAlphaWords=1
ReportDoubledWords=0
CaseSensitive=1
PhoneticSuggestions=0
TypographicalSuggestions=1
SuggestSplitWords=0
```

WARNING

Although GoldMine Premium offers the user the ***Suggestions*** *option, FrontRange does not recommend that the user modify this option at all. It has been set to allow for the optimal performance of GoldMine as its default setting.*

The next frame of concern under this tab is the **Suggestions** frame. There is but one option to set under this frame. The possible radio selections are:

	[SSCE User]
○ **Fast but less accurate**	MinSuggestDepth=25
◉ **Moderately fast and accurate**	MinSuggestDepth=50
○ **Slow but accurate**	MinSuggestDepth=75

The default setting is the ◉ **Moderately fast and accurate**, however, the user may choose to opt for better accuracy. This selection determines the size of the resulting suggestion list based on the users **Spelling Options**. The resulting changes to the UserID.ini are shown adjacent to each radio button option.

The last frame on this tab is the **Dictionaries** frame, and it is here that the user may select from any one of eighteen different dictionaries to use as your **Main Dictionary language:**. Your users would simply select the dictionary of choice from the drop down list. The default dictionary for my installation was **American English**, however, this was based on my type of installation. The UserID.ini controlled by this selection is:

```
[SSCE User]
MainLexFiles=ssceam.tlx,ssceam2.clx
```

In this frame, there is also a button **Setup Dictionaries...**, and this is to allow one to work with different personal dictionaries. The main use here is for the user to be able to modify their personal lexicon file which, in my case, is the file named **DJ.tlx**. The user may add, delete, import or export words to the selected dictionary. Additionally, the user may add different dictionaries, and maintain them here as well.

Login

The **Login** tab, as shown on the next page in Figure 3-44, is only available to those users that possess Master Rights. It is not hard to realize, any changes made under this tab, will affect all users of GoldMine. These settings and options, then, are not stored in the UserID.ini, but instead are stored in the GM.ini file.

WARNING

One of the most common questions asked by GoldMine users is, "How come GoldMine is always trying to login as MASTER?". Always, it is the name of the last user having Master Rights who clicked on the ***Login*** *tab, and then clicked on the* ***OK*** *button anytime after that. Doing this, whether they made changes or not, adds the line* ***User=SOMEONE*** *to the* ***GM.ini*** *file. All network users from that point forward will default to that login name. This instruction to GoldMine can only be removed employing NotePad, and can never be removed through the GUI.*

The first three settings will be preset for you based on your installation. GoldMine will default these settings for the users initially. The GoldMine Administrator may only need to change these settings if they have moved GoldMine to a new server, for instance. All of these settings, or rather the results of these settings are reflected in GoldMine under the **Help** | **About GoldMine...** | **System** button option.

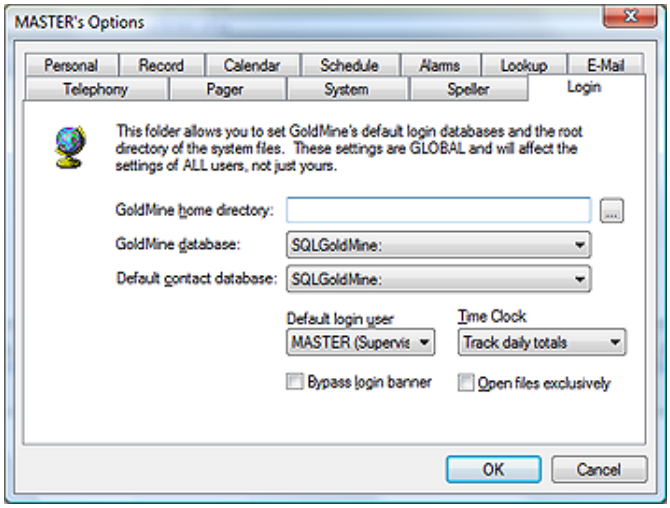


Figure 3-44

The next setting is the **Default login user** field, which defaults to the currently logged in UserID. If the GoldMine Administrator clicks on the **OK** button for these **Options**, any time after being on this tab, that users login UserID will be saved to the GM.ini file. Please read the sidebar **WARNING**.

The **Time Clock** setting is also available on the **Login** tab. This is a drop-down list that has but three choices. The GoldMine Administrator may choose to **Disable the clock**, to **Track daily totals**, or, lastly, to **Track each login**. The default setting, as shown in Figure 3-44, is to **Track daily totals**. I discussed the **Time Clock** tab when I talked about setting up new users in the GoldMine environment. I have seen very few corporations, if any, use this feature. Most, however, forget to **Disable the clock** here forcing GoldMine to continually build, and to maintain unused logs. The three settings speak for themselves.

☐ **Bypass login banner** is not selected in the default configuration. Should your users not be using passwords in GoldMine, this setting will cause them to pass right into, and through the login splash screen, without stopping. GoldMine should start up exactly where the user last left their GoldMine configured (cross your fingers). I have found that this option, in GoldMine Premium, is pretty much ignored at login time in favor of the settings of the shortcut. Let's say that a user has a password, the shortcut could be configured:

```
G:\GoldMine\GMW.exe /u:UserID /p:Password
```

The user, clicking on this shortcut, would be teleported directly into GoldMine without stopping at the splash screen.

The last option, on the **Login** tab, is to ☐ **Open files exclusively**. In a network environment, no one should ever have cause to select this option. Selecting this option, though it may speed up access time for the single user that first logs in, will lock out all other users from the selected GoldMine database. This is no longer a recommended setting for either the Network or Stand-a-lone installation as it no longer has any effect. It may have been best if FrontRange had simply removed this option all together.

Note

Our old timers will recoginize that this is a significant change over GoldMine Corporate Edition 6.7 for instance.

GoldMine Corporate Edition 6.7:

GoldDir=MSSQL:GoldMinePE:dbo:

GoldMine Premium:

GoldDir=GoldMinePE:

The following are some of the possible settings that could be present in the GM.ini file based on entries made on the Login tab:

```
SysDir=Y:\GoldMinePE\
GoldDir=GoldMinePE:
CommonDir=GoldMinePE:
User=DJ
UserLog=1
Exclusive=1
```

Global System Settings

Before we move into our GUIless Settings, as we have in previous editions of this book, we are going to discuss a new GUI incorporated into GoldMine that will prevent the need for some of those GUIless Settings that we discussed in the past. This option can be found off of the GoldMine menu if the logging in user has Master Right.

Tools
Configure ►
System Settings

WARNING

Both of these settings must be in the form of a UNC path, and you may not utilize mapped drives.

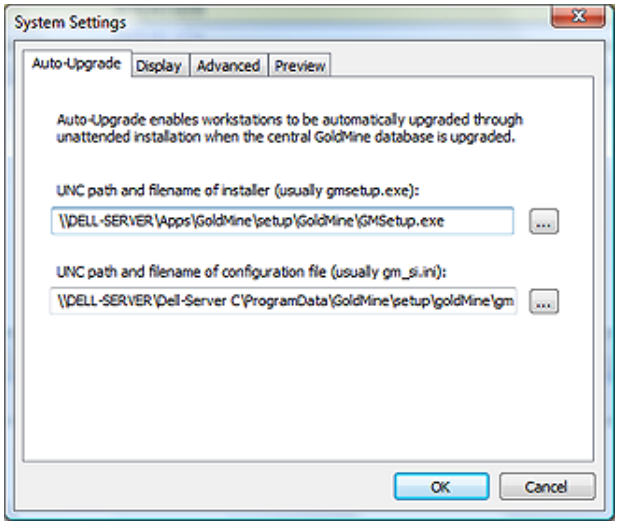


Figure 3-45

The **Auto-Upgrade** tab is the first tab to display. It is on this dialog form that you may establish the paths to the various components that will permit Workstations to be automatically upgraded via an unattended installation after the Network database has been upgraded.

The first such path that must be defined is the **UNC path and filename of the installer (usually gmsetup.exe):**. This is usually under the \\Server-Name\Apps\GoldMine\Setup\GoldMine folder, but not always. Hence, you have the option to designate the true path.

And the second path is the **UNC path and filename of configuration file (usually gm_si.ini)**, but again not always. If you utilized the defaults, then that will be the file name, however, only you know for certain where you asked to have it saved. If you accepted the defaults during the initial installation, then it could be in \\Server-Name\C\ProgramData\GoldMine\Setup\GoldMine\GM_SI.ini.

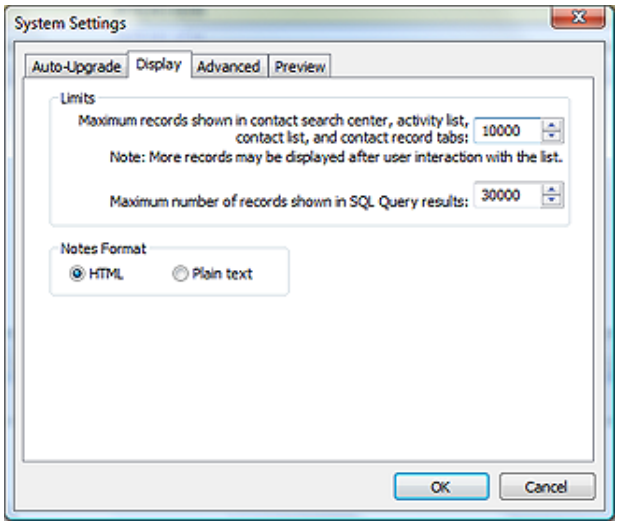


Figure 3-46

Moving forward to the **Display** tab, we can see 2 frames, **Limits** and **Notes Format**. In the prior books the **Limits** options were taken care of as GUIless settings as shown here:

■ Display Limitation - F2 Look-up list display override

In the default configuration of the F2 Lookup list, the list is limited to displaying 1000 items. I had one client that wanted to show some 1250 items in their list (read all about the F2 Lookup list in a later chapter). The UserID.ini came to the rescue with the use of this statement:

```
[GoldMine]
MSSQLMaxBrowseRecs=2000
```

As simple as that, the problem was solved. Again, however, I had to insert this statement into each and every UserID.ini on their network. This is yet another case for the use of the Corporate Override that I will discuss later as part of the GM.ini. What a hassle, as there is no GUI within GoldMine to assist the GoldMine Administrator with this task. All of these edits must be done using a utility application like Microsofts NotePad.

■ Display Limitation - SQL Query tables display override

Later in this book, I plan to demonstrate SQL Queries, and how to output their results to Word, Excel or the Clipboard. In the default configuration, GoldMine will produce a query result table (cursor) containing up to 10,000 records. What if you had 18,000 contacts in your database, and you wanted to send that information to Excel, which would then need to be forwarded to your mailing house? You might be able to export the information from GoldMine, however, if it were any information other than that which is contained in the Contact1 and Contact2 tables, you couldn't export it using GoldMine. Let's say you want Additional Contacts, and you want their E-mail Address as well as the Primary Contact with their E-mail Address. The default SQL Query limit of 10,000 is way too low for most users of GoldMine, why don't we just raise that query limit to say 30,000 records. That should suffice for this job anyway. You can raise the limitation yourself.

This would be the UserID.ini setting to raise the SQL Query limit to 30,000 records:

```
[GoldMine]
SQLQueryLimit=30000
```

Today, however, these are handled in the GM.ini via a **User Override**, when using the **System Settings** | **Display** GUI. The settings displayed in Figure 3-46 are represented here:

```
[User-Override:GoldMine]
SearchTopRec=10000
MSSQLMaxBrowseRecs=10000
SQLQueryLimit=30000
```

In the **Notes Format** frame you have but one option. The option is to have your Calendar and History Notes as ◉ **HTML** or as ○ **Plain text** notes. This, also, was previously taken care of via a GUIless setting in the GM.ini, but today is taken care of via the **System Settings** GUI:

```
[GoldMine]
HTML_Cal_Notes=1
```

Again, we can move forward to the **Advanced** tab which contains two frames the **Contact's permissions** frame, and the **Universal Search** frame.

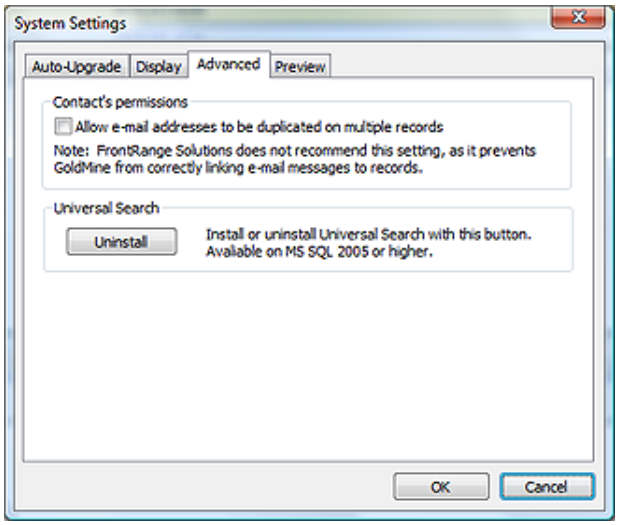


Figure 3-47

As always, we begin at the top and work our way down so the first frame to be discussed is the **Contact's permissions** frame which has a single option: ☐ **Allow e-mail addresses to be duplicated on multiple records**. This is yet another GUIless option that has received a GUI for manipulating its entry. From my previous book:

■ Contact Record - Permitting duplicate e-mail addresses in the ContSupp table

In older versions of GoldMine, GoldMine used to be able to accept a duplicate e-mail address in the ContSupp table. Somewhere along the way that feature got waylaid. GoldMine, in the default configuration, will only accept one instance of an e-mail address in the ContSupp table. The users wanted the ability to have the same e-mail address associated with different contact records, hence, the inclusion of an organization wide ini setting. FrontRange allows for the placement of the following statement in the GM.ini file:

```
[GoldMine]
AllowDupEmails=1
```

The 1 tells GoldMine to allow duplicate e-mail addresses, while a 0 in this statement would return GoldMine to its default state.

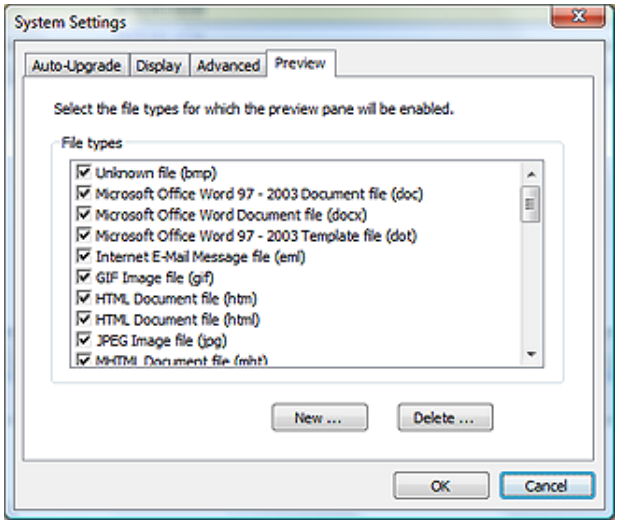


Figure 3-48

Well then, let's move on to the **Universal Search** frame. If you did not install Universal Search when you were upgrading or installing your GoldMine Premium 8.5 then this is the place that you would do this. As you can see in Figure 3-47, I already have Universal Search installed, hence, the button now reads **Uninstall**. What can I say? Click the button.

Finally, we have reached the last tab, Figure 3-48, that being the

Note

*It might be interesting to note that the single setting for the **Maximum records shown in contact search center, activity list, contact list, and contact record tabs:** via the GUI will translate to the two statements in the override:*

```
[User-Override:GoldMine]
SearchTopRec=10000
MSSQLMaxBrowseRecs=10000
```

WARNING

I strongly recommend that you be aware of the Note that FrontRange has added to this frame option, so much so that I am going to repeat it here:

Note: FrontRange Solutions does not recommend this setting, as it prevents GoldMine from correctly linking e-mail messages to records.

Note

*To Install Universal Search, one must first have **Full Text Search** capabilities established and running within **SQL Server**. Without that, you will not be asked to Install Universal Search nor will the Install button, Figure 3-47, be enabled for use.*

Preview tab. It is here that the Administrator may designate the varies File Types for which the GoldMine Preview panels will be enabled. In the **File types** frame is a list of types, and all, in their default state, are selected (checked). It would be up to the administrator to check or uncheck any for which they do or do not want the Preview panel enabled.

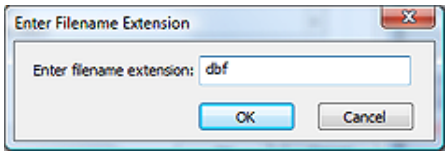


Figure 3-49

Additionally, the Administrator may add any **New...** or **Delete...** any existing extensions. For instance, I added the dbf extension to my list by simply clicking on the **New...** button, and entering the extension into the **Enter Filename Extension** dialog form as shown here in Figure 3-49. A simple click on the **OK** button, and the extension is added to the list and selected upon creation. I mean, why add it if you don't want it enabled?

A simply click on the **OK** button on the **System Settings** dialog form saves all of the changes to the various ini's. These particular settings are entered in the GM.ini similar to:

```
[PreviewPane]
FileTypes= bmp; doc; docx; dot; eml; gif; htm; html; jpg; mht; mhtml; mpd; mpp; mpt; mpw; mpx;
           msg; oft; pdf; png; pps; ppt; pptx; rtf; txt; url; xl; xla; xlc; xlm; xls; xlsx; xlt; xlw; xml; zip;
           dbf;
CheckedTypes=1;1;1;1;1;1;1;1;1;1;1;1;1;1;1;1;1;1;1;1;1;1;1;1;1;1;1;1;1;1;1;1;1;1;1;1;1;1;1;1;1;1;1;
```

Note

*This is probably obvious, but the extension should be a registered extension on the operating system. For instance, when I added the dbf extension, GoldMine picked up the relationship from my system of ☑ **Microsoft Visual FoxPro Table file (dbf)**. Well I am a Visual FoxPro developer after all.*

Note

*Please note that the **[PreviewPane]** line wrap shown here would appear as a single line of text in the GM.ini within the 256 character statement limitation.*

GUIless Ini Statements

Note

FrontRange is constantly enhancing the GoldMine product. A lot of these enhancements include the addition of user options through the use of the UserID.ini. The user, and GoldMine Administrator should spend time in the GoldMine Forums to keep up with the latest information about GoldMine.

Okay, now let's see if I can remove all of the GUIless ini statements from my book that are now handled through the new GoldMine Premium GUIs. There are some entries to the UserID.ini file, as well as the GM.ini file that can not be made through the use of the GoldMine GUI. These entries must be made by hand to the appropriate ini file, and I suggest that the GoldMine Administrator making these changes use Microsoft NotePad. I have tried to cover some of the more popular statements that can be entered into the UserID.ini or GM.ini. I certainly could not cover all of the possibilities as they are constantly in flux. As a GoldMine Administrator, you would be wise to review all of these possible settings, as I'm positive that you'll find a couple that will meet your corporate paradigm.

■ Contacts Additional - Defaulting the Additional Contacts to use the Address information of the Primary Contact record

This addition has been a long time in coming. In fact I wrote an application 10 years ago to facilitate the addition of **Additional Contacts** to the Primary record. Now my Additional Contacts application has been made obsolete with this new UserID.ini entry, and, of course, you can use the corporate override in the GM.ini to set this for all network users.

```
[GoldMine]
AutoFillAddlAddress = 1
```

This is equivalent to selecting the ☐ **Auto-fill with Primary Address** you see when you are creating a New Additional Contact within GoldMine. This is a sticky setting for each UserID, and in that there is this GUI option within GoldMine that can change the setting, if you wish to maintain this setting in any state, you should consider a User Override.

■ Contact Record - Duplicate checking when creating New Contacts

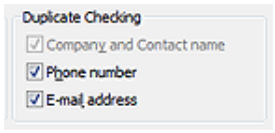


Figure 3-50

When creating a **New Contact** record in GoldMine there is a **Duplicate Checking** frame with options. These settings have changed drastically. Refer to Figure 3-50 for the Duplicate Checking frame as it appears today. These are sticky options such that the user could set them here, and this would maintain the same settings the next time that the user was to create a New Contact record or a New Company record. I thought, however, that you would be interested in various settings in case you decided to use your ability to do a User Override. First I must tell you that the statement in the UserID.ini that controls which of the options are selected, and which of the options are not is:

```
[GoldMine]
NewRecordOpt=4336
```

This statement, as shown, is the default setting within GoldMine Premium, however, being that it is the default it will not be displayed in the default UserID.ini when the user first starts GoldMine Premium. Here are the various possible settings:

Company & Contact	**NewRecordOpt = 4288**
Company, Contact & Telephone	**NewRecordOpt = 4304**
Company, Contact & E-mail Address	**NewRecordOpt = 4320**
Company, Contact, Telephone & E-mail Address	**NewRecordOpt = 4336**

Contact Record - Microsoft Outlook E-mail as a GoldMine History Activity

In the default configuration, whenever a user chooses Send Outlook Message..., GoldMine will create a History activity. This activity will contain no information from within the E-mail message itself, but will only indicate, E-mailed via MS Mail. This is only an indication that the activity was performed. The user, or the GoldMine Administrator, may wish to turn this feature off. The UserID.ini statement for turning this feature off is:

```
[GoldMine]
MSMailHistory=0
```

E-mail Address - Automatically copy the Domain of the Primary E-mail Address when creating a New E-mail Address

I know many have asked for this, however, I don't see it as being particularly useful. With this ini statement, when a user selects to create a New E-mail Address, everything from the Primary E-mail Address Domain, including the @ sign, is automatically completed for the user in the E-mail Address field. From there, the user just needs to add the user name associated with that E-mail Address.

```
[GoldMine]
AutoFillEmailAddr = 1
```

E-mail Limitation - To change the limitation on capacity for the E-mail Center folders.

This setting will also affect performance, and may cause performance related issues. Should you or the user wish to increase or decrease the folder capacity of the E-mail Center folders, it may be accomplished with this setting in the UserID.ini:

```
[GM_Mail_Center]
MailListCapacity = 2500
```

E-mail Center - Performance settings

1. When messages are deleted in the E-mail Center, they are not immediately deleted from the server until the connection is closed. This behavior provides the fastest performance, but can cause deleted messages to reappear as duplicates in some scenarios. If this problem appears, the following UserID.ini setting can be used:

   ```
   [Internet]
   PopImmediateDelete=1 ( closes server connection after each message is deleted )
   ```

   The default value for this setting is 0, which keeps the connection open after deleting a message.

2. GoldMine automatically sends periodic messages ("no-ops") to the e-mail server to keep the connection open for fast response time. By default, these messages occur every 10 seconds. Some e-mail servers require more frequent messages to maintain the connection. To change the frequency of these messages, use the following UserID.ini setting:

   ```
   [Internet]
   PopNoopTimer=nnnn ( where nnnn is the number of seconds between messages )
   ```

3. GoldMine automatically disconnects from the e-mail server if it has no related activity for 10 minutes, by default. This is useful, for example, when using both the Online and Auto-retrieve options with an e-mail server that permits only one connection. To change the time delay before automatically disconnecting, use the following UserID.ini setting:

   ```
   [Internet]
   PopIdle= nnnn ( where nnnn is the number of minutes until disconnect )
   ```

E-mail Center - Change the default refresh rate

In older versions of GoldMine, prior to GoldMine 6.7, GoldMine had a fixed refresh rate. With the release of GoldMine 6.7 the user was permitted to change the refresh rate by hand entering this statement into their UserID.ini:

```
[Internet]
FolderRefreshDelay=60000
```

This setting may or may not have any effect in GoldMine Premium.

Note

*Please note that the user must enter the **FolderRefreshDelay** in milliseconds. The minimum **FolderRefreshDelay** setting is **60000** milliseconds which represents a time delay refresh of **1 minute**.*

Note

*Please note that this setting is **sticky**, hence, if the user changes this while in GoldMine, that change will be reflected in the UserID.ini for the next time. To circumvent this, you may want to consider a User Override.*

■ E-mail Merge - Contact Selection

The default E-mail Merge Contact Selection setting is ⦿ **This contact:**. To have the default be the ○ **All contacts linked to** option or the ○ **All contacts in the following filter/group:** option the UserID.ini section, the setting would be:

```
[EmailMerge]
ContSel=0 ( This contact: )
ContSel=1 ( All contacts linked to )
ContSel=2 ( All contacts in the following filter/group: )
```

This section may or may not exist in the UserID.ini. If it does not exist in the particular UserID.ini simply add the above in its entirety. If it does exist, then just add the statement in the appropriate order.

Note

*Again, you may want to consider a User Override for the **Delivery** frame option.*

■ E-mail Merge - Queue for delivery as default instead of Send E-mail

The default E-mail Delivery setting is ⦿ **Send now**, and many GoldMine Administrators find that to be just plain wrong. Too many e-mail blasts go astray when this is the default setting. One is often better enforcing the ○ **Queue for delivery** option, and having your misapplied e-mail blasts going to the queue rather than going out to the clients directly.

```
[EmailMerge]
Delivery=1
```

This section may or may not exist in the UserID.ini. If it does not exist in the particular UserID.ini simply add the above in its entirety. If it does exist, then just add the statement in the appropriate order.

WARNING

Should you set this option to display AP flow, you want to make certain that you remove the option when you are done with your AP testing. This has been known to cause processing issues..

■ Programmers - Display AP Flow within the GoldMine Process Monitor

There has been a need to debug an Automated Process for a long time now. To facilitate this, FrontRange has added the ability to switch On/Off the flow of the processes through the GoldMine Process Monitor. This is done individually within the UserID.ini. This is the switch statement:

```
[GoldMine]
APDebugLog=1
```

This is a typical result of said action:

```
0[1] Automated Processes  [1:12 pm - 6/24/2009]
0[1] Scanning Contact: Computerese; DJ Hunt
0[1] Read Track:  Update Department Field  [Next Event: 100, Track: 9G4RA45$W=98R#Y]
0[1]    100   Test Update Field
0[1]            --> Triggered.
0[1]                Update Field: DEPARTMENT with DJ Hunt
0[1]     Remove ended track: 9G4RA45$W=98R#Y
4[1] Automated Processes: 1 Contacts; 1 Scanned; 1 Triggered; 1 Pass(es)  [Ended 1:12 pm -
6/24/2009; Dur: 0:00]
```

As you can see this could be valuable information when trying to debug an Automated Process. Of course, if you are perfect, and never make a mistake when developing your AP's this is, naturally, not required.

■ Programmers - Option to trace DDE issues within the GoldMine Process Monitor

How many of you have developed an application that uses DDE to converse with GoldMine, and it does not. The conversation never takes place that you are aware of. You could put this statement into your UserID.ini to allow you to trace, within the GoldMine Process Monitor, all of the DDE conversations that were processed by GoldMine.

```
[GoldMine]
DDEMonitor=1
```

Having put this into my DJ.ini, I was able to follow the conversation shown below within the GoldMine Process Monitor window. I copied it to the clipboard, and pasted it here for your review. In your case that might resolve to printing it for analysis.

```
0[1] DDE Monitor  [3:04 pm - 6/24/2009]
0[1] DDE In : &Version
0[1] DDE Out: 8.5.1.6
0[1] DDE Monitor ended  [Ended 3:04 pm - 6/24/2009; Dur: 0:00]
0[2] DDE Monitor  [3:04 pm - 6/24/2009]
0[2] DDE In : &SysDir
0[2] DDE Out: c:\goldmine\
0[2] DDE In : &GoldDir
```

```
0[2] DDE Out: MSSQL:SQLGoldMine:dbo.
0[2] DDE In : &CommonDir
0[2] DDE Out: MSSQL: SQLGoldMine: dbo.
0[2] DDE In : &UserName
0[2] DDE Out: DJ
0[2] DDE Monitor ended  [Ended 3:04 pm - 6/24/2009; Dur: 0:00]
```

■ Reports - To Snap to Grid or not

This is one of my personal peeves. Whenever I would go into a report in layout mode, I would first have to remember to right-click, select Grid Settings, and to then unselect ☑ **Snap to Grid**. Also, I can't tell you how many times that I forgot to do that, and totally messed up a report that took a long time to create. Until, that is, I found the UserID.ini setting to set the snap to grid option to default to unselected. That option is:

```
[GraphObj]
Grid=0
```

Let me tell you, right here and now, this one setting has saved tons of headaches and frustrations. One setting is worth a thousand screams of frustration.

■ Scheduled Activities - Private automatically selected when scheduling activities

A similarly functioning setting to the **OnByDefault** setting, in the UserID.ini, is the ability to have the ☑ **Private** option selected by default when scheduling a new activity. This is, again, set by using the RecType for the activity to be designated as private.

```
[ActvObj]
PrivateActvity=ACTDO
```

I must point out the spelling of the statement above. Even though this may appear to be a typographical error, it is, in fact, correct. Adding the i to Activity would cause this statement to not function. The reader is reminded that they may change the instructions on the right side of the equation, but not on the left side.

Note

*Please note, as of this writing, the **M**, message option, does not function in the **PrivateActvity** statement. Additionally, there can be no private selection for the **S**, sales option.*

■ Tabs - Renaming the default GoldMine tabs

Here is a little feature that the FrontRange people shy away from, for what I believe should be, obvious reasons. How many times have you been asked, "Can't we change the name on this tab to something more meaningful to our organization?", and of course it has always been one of the default tabs that they wanted changed. Well there is a way within the UserID.ini, although, if you are going to do this, I would recommend that you use the Corporate Override. Let's say that you wanted to change the tab named Notes to now display Memos, this is how that would be accomplished via the UserID.ini:

```
[GoldMine]
ROTabs1=&Summary,&Fields,GM+&View,&Notes,Additional&Contacts,&Details,&Referrals,&Pend-
ing,&History,&Links,&Members,Pr&ocesses,Opport&unities,Pro&jects,Relati&onships,C&ases,AR_
Aging,
```

If this is a corporate naming convention, the GoldMine Administrator will need to insert this statement into each of the UserID.ini files, one at a time unless you opt to use the User Override which I will discuss later in this chapter. Also, should you rename any tab, do not call FrontRange for technical support, and then say my Memos tab is not functioning properly. They will not have a clue as to which tab you may be talking about, and will not be able to assist you with your problem. This applies to any of the GoldMine default tabs that you, or your organization, may wish to rename.

Note

*In the **ROTabs1** statement the user, or GoldMine Administrator, must supply a name for each of the GoldMine default tabs. If this is not done, then those tabs for which a name was not supplied, will still remain, but will have no label identifying their contents.*

Note

*Did you notice the **&** as part of some of those tab names?*

&Summary,&Fields, GM+&View

*These **&**'s designate a hot key for quickly moving between tabs. Be careful, if applying your own hot keys, as they may conflict with already defined hot keys.*

■ Warnings - Turning On/Off Various Warnings within GoldMine

GoldMine warns you about deleting an E-mail, but then offers a checkbox to disable any future warnings. There is no GUI to turn this feature back on. You must do this by modifying the UserID.ini. Here are some of the various warning messages, in GoldMine Premium, that can be manipulated manually.

```
[Warning]
InetWarnAboutDelete1=0
InetWarnAboutRTF=0
BrowseSort=0
ORGCHARTDRAGWARN=0
WarnAboutSCSpeed=0
InfoWarnAboutDrag=0
WelcomeRecTypes=0
WelcomeGoldSync=0
```

Note

*Remember that a **0** is **Off** while a **1** is **On**.*

Note

*You may want to consider a **User Override** for these annoying warnings. User Overrides are discussed later in this chapter.*

```
WelcomeInfoCenter=0
WelcomeAutomatedProcesses=0
WarnAboutWebImport=0
WarnChangeMasterPwd=0
WarnAboutImport=0
WarnMakeBackup=0
WarnAboutDeleteRecords=0
WarnCMTemplatesEmail=0
WelcomeInetEmailRules=0
WarnAboutGlobalReplace=0
NotesLimit=0
WarnAboutMergePurge=0
WarnAboutCopyMoveRecords=0
WarnAboutMaintain=0
WarnAboutEMailMergingWhenSMime=0
WelcomeDistList=0
WarnAboutCompleteMultiLinkActiv=0
WarnAboutDeleteMultiLinkActiv=1
WarnAboutEditMultiLinkActiv=1
WelcomeCampaignManagementCenter=0
WarnCMTemplatesWrite=0
WarnCMFilterGroupActivate=0
InetWarnAboutMerge=0
ORGCHARTROLLUPWARN=0
WarnAboutPalmDesktop=0
WarnCMSalesForecast=0
```

GM.ini

The GM.ini has never had a GUI to assist in its creation save for the **Login** tab under the Master Rights users Options, and now some overrides via the System GUI. Through the remainder of this chapter I will be discussing some of the statements that are applicable in the GM.ini. One major feature that I will be discussing is the ability to override the UserID.ini Options via the GM.ini to present a solidified corporate approach to your GoldMine presentation and usage.

■ Additional Contact - Fax Label nomenclature

The Additional Contact has had a **Fax:** label for the associated field for some time. FrontRange has listened to the end user, and they understand that more Additional Contacts are apt to have a Mobile telephone while it is less likely that they would have a Fax. Hence, GoldMine no offers the ability to change the label of this field using the following statement:

```
[GoldMine]
AddContactFaxLabel = Mobile ( or any other label that you may choose )
```

■ Block Execution/Launch - Stop GoldMine from running Attachments via their launch registration

This switch is a rather long one, and could have been longer (up to 256 characters in length). This statement blocks attachments from executing from inside of the GoldMine E-mail Center. All of the listed types of attachments could still be present within the message, however they will not automatically execute/launch from within the GoldMine environment when you double-click on the attachment in the e-mail.

```
[GoldMine]
BlockAttachExecution=asp,bat,com,doc,exe,htm,html,inf,js,jse,lnk,nws,pif,pp,ppt,scr,vbe,vbs
```

■ User Override - Overridding the users Options at the corporate level

Through the GM.ini, you, the GoldMine Administrator, can set a corporate display/functionality presence for your GoldMine. Regardless what the users choose for their Options at their level, you can override these at the corporate level. Although this won't prevent the user from changing their own Options, when next they log into GoldMine the overridden options will be reset to the corporate standard.

Now I can discuss the UserID.ini override options syntax. You now have the option of adding new sections to the GM.ini.

These sections take the form of:

```
[User-OverRide:<User.ini Section>]
```

You may include one override section for each section in the UserID.ini that you may want to override. Here are some of the overrides that I have in our GM.ini for example:

```
[User-OverRide:GMAlarm]
OnByDefault=ACTSOML
[User-Override:GoldMine]
ROTabsGlobalCheck=0
MSSQLMaxBrowseRecs=10000
SearchTopRec=10000
SQLQueryLimit=30000
```

These overrides are a great way for the corporate GoldMine Administrator to enforce corporate standards for their GoldMine.

Here is a override that I discussed earlier in a sidebar Note. This override will establish, at the corporate level, the GoldMine Warnings as they are desired for the corporation. Regardless of whether the user tries to turn them off, they will re-establish themselves the next time that the user starts GoldMine.

```
[User-Override:Warning]
InetWarnAboutDelete1=0
InetWarnAboutRTF=0
BrowseSort=0
ORGCHARTDRAGWARN=0
WarnAboutSCSpeed=0
InfoWarnAboutDrag=0
WelcomeRecTypes=0
WelcomeGoldSync=0
WelcomeInfoCenter=0
WelcomeAutomatedProcesses=0
WarnAboutWebImport=0
WarnChangeMasterPwd=0
WarnAboutImport=0
WarnMakeBackup=0
WarnAboutDeleteRecords=0
WarnCMTemplatesEmail=0
WelcomeInetEmailRules=0
WarnAboutGlobalReplace=0
NotesLimit=0
WarnAboutMergePurge=0
WarnAboutCopyMoveRecords=0
WarnAboutMaintain=0
WarnAboutEMailMergingWhenSMime=0
WelcomeDistList=0
WarnAboutCompleteMultiLinkActiv=0
WarnAboutDeleteMultiLinkActiv=1
WarnAboutEditMultiLinkActiv=1
WelcomeCampaignManagementCenter=0
WarnCMTemplatesWrite=0
WarnCMFilterGroupActivate=0
InetWarnAboutMerge=0
ORGCHARTROLLUPWARN=0
WarnAboutPalmDesktop=0
WarnCMSalesForecast=0
```

■ Display Limitation - Pending Activities (All) in the Activity List

There appears to be a display limitation in the Activity Listing. One user had selected All activities to be displayed, however, they were only able to see activities through January 2007, and they were positive that they had more to be displayed. FrontRange suggested that they add this statement to their GM.ini to resolve the display issue:

```
[GoldMine]
PendingSQLMaxRows=2000
```

■ General Statements - What they are, and what they do

The following are statements from a GM.ini that I would give to my clients that can be modified to their needs and uses. However, it is a beginning point.

```
[GoldMine]
```

There can only be one [GoldMine] section in the GM.ini, and all statements that are applicable to this section must go under this one section header.

The SysDir belongs under this section, and points to the folder where the License.bin resides.

SysDir=Y:\GoldMine

The GoldDir, also under this section, points to the location of the GoldMine (GMBase) database. In my case this is:

GoldDir=GoldMinePE:

While the CommonDir identifies the location of the Contact set database. It is not uncommon for this to be the same location and setting as the GoldDir statement.

CommonDir=GoldMinePE:

Should you want the big brother approach to corporate oversight, you can turn on/off the logging of the users keystrokes, and mouse clicks. As always a value of 0 turns off the option, while a value of 1 turns on the option.

UserLog=0

If you have set up, and use Record Alerts, these will pop up, when you display an alerted record, 7 seconds after you display the record. Personally, I prefer a much shorter interval. In fact, I use 2 seconds, and here is the statement that controls this for me.

RecAlertSec=2

In the old days of yor, GoldMine did not automatically create a Contact2 record for each Contact1 record that was created. This was a space saving feature when space was at a premium. Todays GoldMine creates the Contact1/Contact2 records at the time a new record is created, however, for older versions of GoldMine, this legacy GM.ini statement is still maintained, and it forces GoldMine to create a Contact2 record whenever GoldMine creates a Contact1 record.

NewComplete=1

These are but a few of the statements that can be employed under the [GoldMine] section of the GM.ini.

■ MyGoldMine - Making it YourGoldMine

I have always complained that MyGoldMine is really FrontRanges GoldMine as I had no control over it. John Stillman, of FrontRange, has added some override parameters to the GM.ini where you can now point MyGoldMine to your own RSS feed, or any RSS feed for that matter. This is a totally separate section in the GM.ini, and only functions in GoldMine 6.70.50123 or greater.

[MyGM]
FRS_Title = Title
FRS_URL=http://www.Domain.com/RSS.xml

■ Output Data - Turn On/Off iCal messages

This switch turns off the option to output iCal messages from the Pending tab, Activity List (F6), or the Calendar.

[GoldMine]
AllowiCalOutputTo = 0

■ Output Data - Output Data - Turn On/Off iCal messages

Hand in hand the next switch turns off the Output To menu in the Calendar for iCal and HTML.

[GoldMine]
AllowCalOutputTo = 0

■ Scheduled Activities - Turning On/Off HTML Notes

GoldMine Premium is using Internet Explorer 7 to create and display Notes in History, Calendar, Info-Center, and other areas. Many users have complained about this, especially when reporting against HTML Notes, and FrontRange has decided to add a switch to turn this feature on or off. This switch only applies to Notes in History and/or in the Calendar, and does not affect Notes that were already created prior to utilizing this switch. This is the GM.ini setting to accomplish this:

[GoldMine]
HTML_CAL_NOTES=1 (default)
HTML_CAL_NOTES=0 (turns off the HTML Notes)

- **Speller - Identifies the path to the Speller files**

This statement identifies, to GoldMine, the folder under GoldMine, usually, where the dictionaries are maintained for GoldMine.

```
[GoldMine]
SSCE_PATH=Speller
```

No path is required if the corporation is using the default folder. Many times however, GoldMine Administrators will have these files locally on the workstation for better performance. That activity, and this being the corporate GM.ini, all users must have the Speller folder in the same location on their Workstation, and this pointer must be appropriately set for the corporation.
An example might be:

```
[GoldMine]
SSCE_PATH=C:\GoldMine\Speller
```

- **User Variables - User variables that are common to everyone in the organization**

Global user defined variables can be defined, and employed in merge documents and e-mail messages. Some of these might be the corporate website, the corporate fax, or the main corporate telephone number. This is how you might establish these in the GM.ini:

```
[user_var]
WEB=www.DJ-Hunt.com
PHONE=(978)342-3333
FAX=(978)123-4567
```

- **X-Mailer ID - Sometimes triggers spam filters**

Header information included in GoldMine Outgoing E-mail will sometimes trigger the recipients Spam Filters. GoldMine Administrators may change the course of this action by adding a GM.ini entry that will override the X-Mailer ID.

```
[GoldMine]
Xmailer = Yahoo ( Use any other name here instead of GoldMine )
```

4

In This Chapter

The ability to add user defined fields affords the GoldMine Administrator the ability to add fields to their GoldMine application to meet the needs for their particular organization. A field, as I define it, is a place to store one piece of information. For example, the **Source** field, a default field in GoldMine, allows one to store the original single source through which a particular contact became familiar to your organization. This field is character based, and it is defined to be 20 characters in length. If the contact first came to know of the organization through a trade show, and then later reestablished contact with the organization through a marketing campaign, the Source field would not be the place to store and analyze this type of information. An organization that desired to analyze and track multiple contacts would be better served by using the GoldMine one-to-many capability of the **Details** tab. However, I would be heading off on a tangent with that one, and should follow the straight and narrow discussing **Custom Fields**, **Custom Screens**, **Screen Design** and **Record Typing**.

I have combined these items into this single chapter as they go together, hand-in-hand. One must create Custom Screens in order to present the Custom Fields to the users for data entry. Once created, they could be utilized by the Record Typing feature of GoldMine. In rare circumstances, one may want to create user defined fields, and not place those fields on a screen for data entry or data editing. Sometimes this type of user defined field is auto-populated through use of the **Lookup.ini**, discussed in detail later on in another chapter. Such a field could be a counter field that is auto-populated with a unique number. Most often, however, user defined fields are placed on user defined screens, and if they must have special characteristics, those are handled on either the screen level or the field level. I'll discuss this as I talk about each item individually.

Custom Fields

WARNING

Prior to doing any customizations, you are warned to backup your GoldMine databases. Your only recover from an errant deletion is via a restoration of the database.

You have been warned.

We will now proceed with **Step 1** of my process, that of creating the custom fields. You must create your custom fields, then create your custom screens or visa versa, and finally, meld the two together to create a unique custom display for your information. We can now proceed to Custom Fields, and it is here where we collide with the new learning curve inherent with change. The new location for this GoldMine menu item is:

Tools
Configure ►
Custom Fields...

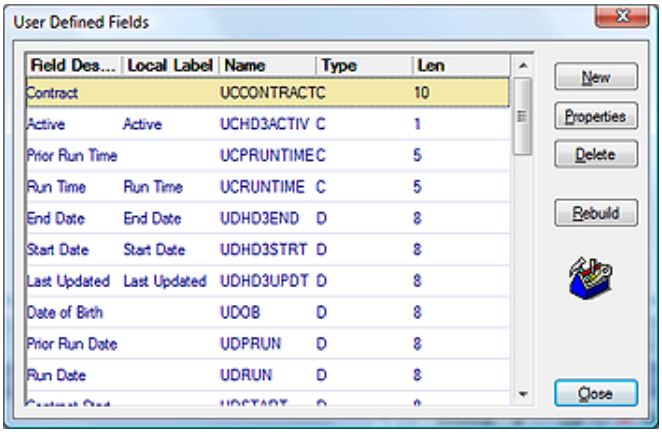


Figure 4-1

This will bring up the dialog form shown here in Figure 4-1. One of the first things that the astute reader will notice, if theirs is a new installation of GoldMine, is that there are 16, yes count them, 16 user defined fields in the GoldMine default configuration. FrontRange has taken the liberty of predefining some of your user defined fields for you. Immediately, I would **Delete USERDEF11** through **USERDEF16** inclusive as they are not so very user

defined now, are they? As for the other 10 user defined fields, which were not defined by any of your users, I usually take them right out of play by reducing their length to 1 character, and removing their description. Oh, and in case you were wanting to give it a try, the deletion of USERDEF01 through USERDEF10 is forboden. I find that they are not anything that I would define, and that they do not follow any convention that I would wish to employ. We'll talk about conventions in a little bit, but first let's eliminate some of those buttons that were shown in the Figure 4-1 dialog form.

The **Delete** button does just that which is stated (go figure), it deletes the selected (highlighted) user defined field from the **ContUDef** list which, when the **Rebuild** takes place, will in turn remove it from the **Contact2** table.

The **Rebuild** button, on the other hand, plays a much larger roll. After one has added, changed the properties of, or deleted fields from the **User Defined Fields**, Figure 4-1, dialog form, GoldMine will ask them to rebuild the tables. This assumes that the user forgot to click on the **Rebuild** button themselves, and has attempted to **Close** the dialog form. Either way, it takes the clicking of this button to add, modify, or remove the definition record from the **ContUDef** table. This is the table where all of the definitions for user defined fields are maintained, the user defined data dictionary if you will. Once this dictionary has been modified accordingly, then GoldMine can proceed to modify the related structure of the **Contact2** table based upon the definitions contained in the **ContUDef** table. Each record in the **Contact2** table will need to have fields added, modified, or removed from them accordingly. Tables, indices and structures will be presented in more detail later in this book.

Note

In the past, I used to warn individuals that they could not add more than 230 user defined fields through this GoldMine dialog form for the GoldMine Premium. Now, although I haven't actually tested it personally, others have, and I am told that you can now add as many fields as you would like here. As I mentioned in the past, this was a BDE limitation, and in GoldMine Premium, with the switch to ADO, this limitation has gone away although there is still a SQL Database limitation of 8,196 bytes per record in any table.

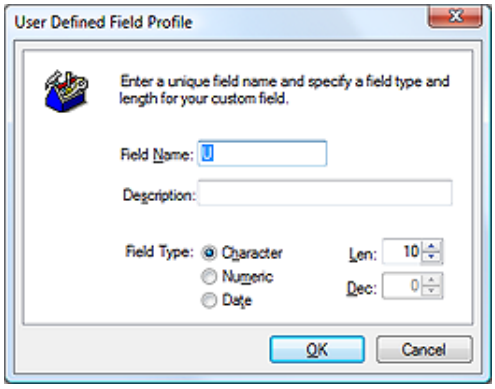


Figure 4-2

The **Properties** button will bring up a dialog form, similar to that shown here in Figure 4-2, however, unlike the **New** button dialog form, which I am showing you in Figure 4-2, all of the information will be contained there on, and it will be ready for any modifications.

Lastly, I will discuss this dialog form as presented to the user when they click on the **New** button. The user will notice that there are a very limited number of properties that they may ascribe to each field. This has always been an issue for GoldMine users.

The **Field Name:** automatically contains a U as the first character in the field. All user defined fields in GoldMine must begin with the letter U. If you overwrite the preassigned letter U, then when you click on the **OK** button, GoldMine will append one to the beginning of the field name for you. User defined field names, in GoldMine, may only be 10 characters in length, and the first character must be U. They may not begin with a special character or a number. Based on that, the user has 9 alphanumeric characters with which they may name their user defined field. I always try to assign some relevant meaning to this name, i.e. a field that is to contain a date of birth might be named, udDOB. This field name is apt to appear on a GoldMine field list at some point in time as you are using GoldMine. This convention (Hungarian convention) lets one immediately know that it is a user defined field (**u**dDOB), that it is a date based field (u**d**DOB), and that its name is ud**DOB**. Would one have a better understanding of what might be contained in a field from that list, if one saw UserDef08, or udDOB? That was a rhetorical question, naturally. Obviously. I don't care for the GoldMine field naming convention, and I suggest that you develop and employ a field naming convention that is meaningful to both you and your end users alike. Your only limits are the 9 character length of the field name, and that you may not employ spaces and/or special characters in the name.

The **Description:** field has a little more significance in GoldMine Premium. When one places the field onto a screen for user data entry, GoldMine will take the first 15 characters of the description as its best guess for its **Global Label**. Before this field is placed on a screen for data entry, in some cases, GoldMine will display the description instead of the field name on that list. I just try to make the description as meaningful as I can within the 25 character **Description:** field limit.

Note

*Alas, no changes in GoldMine Premium. There are still no **BLOb** (**B**inary **L**arge **Ob**ject) user defined field type. It is this field type that one would need to create a user defined **Notes** type of field. This has often been requested by end users, and may find its way into future builds of GoldMine. Additionally, check boxes and radio buttons, which are **Logical** (Boolean) type fields, are not available at this time.*

Ah yes! **Field Type:**, a radio button selection for one of three possible data types:

- ◉ **Character**
- ○ **Numeric**
- ○ **Date**

That's it, there aren't any other types, at least not as of this writing. There have been a multitude of requests for memo field types, or logical field types. I want to say, however, that things that are often requested, usually make it into future releases of GoldMine (of course this is the 7th rendition of this book in which I have made this very same statement). If having additional field types is an issue

for you, and it is for most of us, then you are requested to send your suggestions to: **Suggestions@FrontRange.com**. The more requests they receive, the more likely that we'll see these incorporated into a future release of GoldMine.

Let's examine the default ⦿ **Character** data type of field. This type of field can hold any type of character, number, or special characters. A common mistake is to use a numeric field to hold number characters. When a field will hold phone numbers, or social security numbers, the field should be a character based field. Should you want the field to display currency, $1,345.67, you would need to make the field a character based field. Once you have decided that a particular field is to be character based, you must then designate how many characters the field should be allowed to hold. You would do this, setting the limit, using the **Len:** field. The spinner control associated with the **Len:** field will allow you to spin up or down with the default character length of **10** characters.

One could also choose to define a field as a ○ **Numeric** data type field. These type of fields would, most probably, have mathematical operations performed against them. If you were to set your field to this field data type, you would also be required to define the **Len:** of the field, as well as the number of **Dec:** (decimal places) that the field should carry. When you consider the length for a numeric field, remember to add on an additional place for the sign of the field plus (**+**) or minus (**-**). You must also consider that the decimal itself, if you are having one, will take one space. A number like 1234.67 would require a length of 8 with 2 decimal places. Don't forget Murphy's Law either, try to allow for the possibility that you may have a larger number to contend with in the field in the future.

The last field data type is the ○ **Date** field type. Should you select the field to carry a date value, then the length will be set for you automatically. There are no other settings about which you should be concerned.

This, then, just about does it for **Custom Fields**. There is just not that much more that I can discuss on this issue. Add as many fields as you desire within the previously noted limitation. You can modify them at any time by highlighting them, and then by clicking on the **Properties** button. The same dialog as is shown in Figure 4-2, will be brought up with the current field definition, and you may edit the settings there.

WARNING

*As of this writing, GoldMine Premium does not display all numeric values correctly. For instance: **1.00** will be displayed as **1**. Whereas, **1.90** would display as **1.9**. Trailing 0's are truncated for some reason that elludes me.*

If you need Numeric fields, you are advised to mirror the Numeric field with a displayed Character based field for data entry, and to then have the Lookup.ini (refer to Chapter 6) populate the Numeric field while properly formatting the displayed Character field.

Custom Screens

I now take up the discussion of **Custom Screens**, please do not confuse this with **Screen Design** as will be discussed later in this chapter. **Custom Screens** are user defined screens for displaying **Contact1/Contact2** fields and/or **Expression** fields under the **Fields** tab and/or on their own designated tabs as the designer so chooses. Think of these as blank pieces of paper which one will later cut holes into to expose certain data fields to the end user for data viewing/entry/modification.

Tip

*Users not possessing Master Rights will still see **Custom Screens...** and **Custom Fields...** visible on their menu even though these selections will not function for these users. The GoldMine Administrator would be advised to remove these selections from a non Master Rights user menu.*

One must be a user possessing Master Rights in order to select

Tools
Configure ►
Custom Screens...

from the GoldMine menu. A user possessing Master Rights will also be able to select Screens Setup... from the local menu when they right-click in the screen area under the Fields tab. Either of these selections will bring the user to the **Custom Screens Setup** dialog shown here in Figure 4-3, although, for a new GoldMine Premium installation the default screens will cause the **Custom Screens Setup** screen to appear different than that which is displayed here. It is from this dialog form that the user has the ability to change the **Properties** of an existing screen, add or **Clone** a **New** screen or **Delete** an existing screen. Each of these methods has a button except for the **Delete** method. The **Delete** method must be accessed through the **Local Menu** (right-click in the area under the heading row), or via the keyboard, **Delete** key, and will act upon the highlighted record on this dialog form. As I state in the sidebar **Note**, you will not be able to delete the currently active screen.

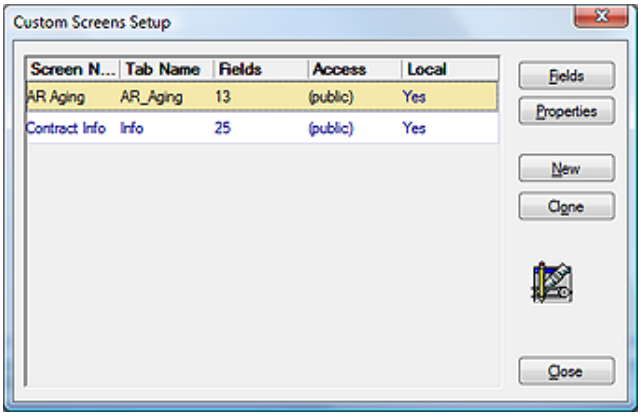


Figure 4-3

Note

GoldMine will not allow the active screen to be removed. There must always be at least one user defined screen even if you don't plan on using the screen for your organization.

Tip

*This Tip may or may not hold true in GoldMine Premium 8.5.1.12, however, remains a good tip just the same. Screens on the GoldMine local menu, under the **Fields** tab, will appear in the order in which they are created, and will not be in alphabetical order, for example. I always create my screens without regard to order, and, when finished, I clone them in the order that I want them to appear on the local menu list. After I am through with my cloning process, I simply remove those screens from which the clones were derived.*

GoldMine Premium now has five screens created in the default installation, the **End User** screen, the **Car** screen, the **Service Customer** screen, the **Other** screen, and the **Prospect** screen. None of which are particularly useful to most organizations, and most will delete them, save one, which they

Note

GoldMine will permit the addition of 19 user defined screens for a total of 20 user defined screens. Of those, only up to 18 may be defined to have their own tab. The 18 user definable tabs must be shared between user defined Screen tabs, and user defined Detail tabs.

will modify. I had started by highlighting the **Car** screen, right-clicking on it, and selecting **Delete...** from the local menu to remove it. I then continued the process until all of the screens had been removed except for the **End User** screen. I next highlighted the **End User** screen, and I clicked on the **Fields** button.

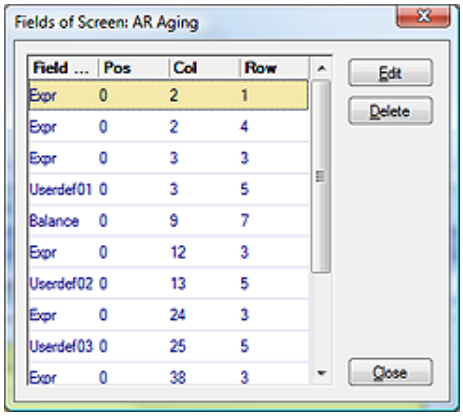


Figure 4-4

This brings up the **Fields of Screen:** dialog form similar to that shown here in Figure 4-4. About the only use that I found for this dialog form is to power delete fields from a screen. That is what I would recommend doing in this case. No organization will have any use for the screens as they come from the GoldMine default installation, although they are good examples of possibilities within GoldMine Premium. I do not recommend trying to layout out screens using this dialog form, as I prefer to visually see the layout as I am designing it. Highlighting the field to be removed, and clicking on the **Delete** button will do the trick. The user should hover over the **Y** key on the keyboard, and get into a rhythm of clicking on the **Delete** button, and then the **Y** key to remove the field from the screen. I always remove all of the fields, and I then begin with a new empty slate (screen). When I am through deleting all of the fields, I click on the **Close** button to return to the **Custom Screens Setup** dialog form.

Note

*Unlike past releases of GoldMine, in the **Tab Name:** field you are now permitted to utilize all 10 character positions, and all of those utilized will appear on the tab.*

*Additionally, whereas previously one had to utilize the underscore (_) to concatenate words for the GoldMine tabs, todays GoldMine will permit the use of spaces between words. Whereas, in the past we would have entered **AR_Aging**, today we can utilize **AR Aging** for a much more user friendly display.*

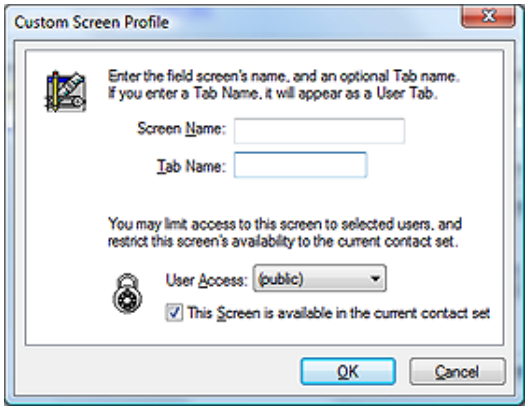


Figure 4-5

Next we should look at the dialog form presented when one clicks upon the **Properties** button as shown here in Figure 4-5. You may assign whatever name you desire in the **Screen Name:** field. This is the name that will appear on the local menu when the user right-clicks under the **Fields** tab. The next field, the **Tab Name:** field, on this dialog form is where one would assign this screen to a tab. To assign a screen to a tab, simply give it a name in this field.

Remember those **User Groups** that I discussed with you back in Chapter 2, well the **User Access:** field is one spot where you could employ a **User Group**. You may limit access to a screen to an individual GoldMine UserID or, to a group of GoldMine UserIDs. One must remember that this setting will not limit access to the screen, if the logged in user has Master Rights within GoldMine. Once access is limited, this screen will not even appear on the local menu as a possible selection choice unless the logged in GoldMine UserID is a member of the specified **User Group** that was given access rights. Should the screen have been defined as having a tab, that tab, as well, will only be available to those having access rights.

☑ **This Screen is available in the current contact set** is selected by default. One could easily wonder why this would even be here as an option. This screen information is maintained in the **Fields5** table which is common to all contact databases. This screen will appear in any newly created contact database. Though one may desire this screen in one particular contact database, they may not require it in another contact database. The GoldMine Administrator is **required** to edit this option for each screen in each contact database. Obviously, if you would want this screen in each contact database on your GoldMine system, then you would not need to edit this option. Editing is only required if you wanted to remove a particular screen from a particular contact database.

Once back at the **Custom Screens Setup** dialog form, you may click on the **New** button to add as many blank screens as you may have need for in your organization. It is usually considered better to have smaller screens of clustered information than one screen containing as many as 250 user defined fields.

The **Clone** button will exactly clone one screen into another, even containing the same name. This is best employed when trying to order screens for presentation to the user (see tip sidebar on previous page).

Now that you have your canvas (the screen) as well as your paints (the fields), you could begin to paint your picture. As I stated previously, I prefer to layout my screens graphically, although I must

warn you that this becomes a much more daunting task in GoldMine Premium. Large screens that used to take me 2 hours, can now take 3 and 4 hours to develop because of the idiosyncrasies within GoldMine Premium. Having said that, to do this, you would click on the **Fields** tab to bring them into focus. You would then right-click in the screen area under the tab, and select the screen that you would like to work on. Should you have assigned a tab name to a particular screen, you could go directly to that tab, and do your screen design in the area under that tab. In either case, once you are on the appropriate screen, you would right-click, and then select **Screen Design...** from the Local Menu. This will place your screen in design mode, showing row lines to aide you in field positioning. A toolbar will display itself as well. The toolbar will not appear in your design area as I show it below in Figure 4-6. Instead, it will position itself off of the screen while remaining easily accessible (maybe, I just had a hard time finding my toolbar). The toolbar icons, from left to right, permit you to place a **New** field, to **Save as** a Primary Field view, to **Load** a Primary Field view, to **Rebuild** to include new fields or to **Exit** the designer.

Note

*GoldMine Premium has the option to Customize the Designer toolbar, **NOT**. Even though the tools are there, the functionality is not present.*

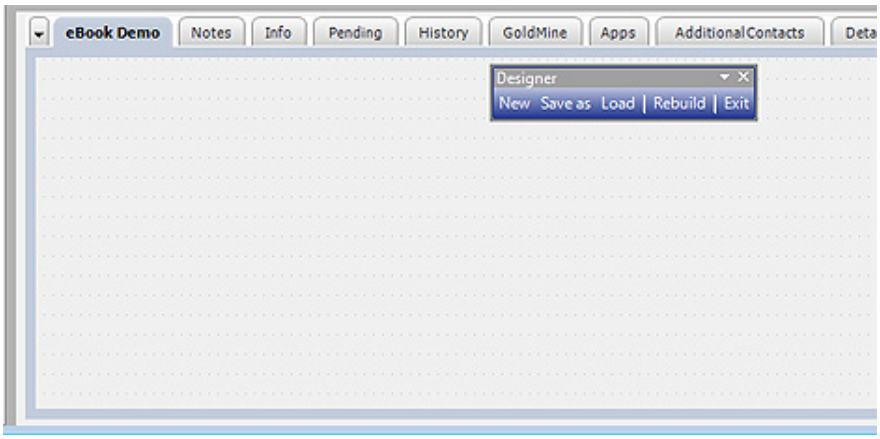


Figure 4-6

The first type of field that I am going to insert is an **Expression** field. This type of field is used to display information only. The user will not be permitted to enter anything into this field as it is not a container (data input) field. We are not limited to the 15 characters that are allowed for labels of container fields either. In some cases, where I wanted an entire question to appear on my clients screen, I would add the question as an expression field, and then after it, add the container field without any label. You'll learn these various techniques as you get a few screen designs under your belt.

Note

*Even though the fields in the top portion of the GoldMine screen were initially laid out by FrontRange, you may change the attributes of the fields if you have Master Rights. Hold the **Ctrl** key down, and double-click on the field for which you wish to change the **Field Properties**. The result will be the dialog form, as shown in Figure 4-7, for the field you have selected. If you click on the **More options...** button, you will bring up the dialog form shown in Figure 4-8, which used to be the **Field Properties** dialog form, but today is known as the **Advanced Options** dialod form. From here you may change the label, the access rights, or any of the properties for these designed fields. One of the most common changes in this area is the **Field Access** properties.*

Remember that this is not changing the actual field properties in any way, and that this action is only changing the properties for the display of the field information.

I'll show you a screen shot for the expression field **Field Properties** selection, however, I'll talk about this tabbed Field Properties screen in more detail when I add the **ucSSN** field in a moment. For this, an expression field, look here at Figure 4-7. This is a change from past versions of GoldMine in that we must first give this expression field a **Label used for 'GM Dealer' record type:**, and, albeit disabled, the **Name in Database:** is displayed. Although, this is not signigicant for what we are currently discussing, it will play an important role when later we discuss **Record Typing**. For the time being, let's just click on the **More options...** button to bring us to, what is now titled, the **Advanced options** dialog form, refer to Figure 4-8 on the next page. In previous books this dialog form would have had the title: **Field Properties**.

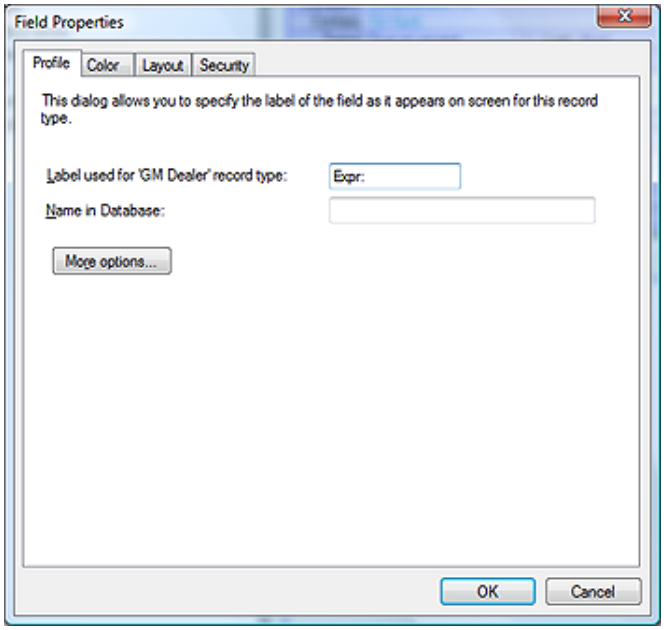


Figure 4-7

For this writing, I called this screen **Personal**, and it will contain personal information, in this particular case the **Social Security Number**. As one could have multiple screens represented under the

Note

It is a coding rule, when you are concatenating things together, that they must be all of the same type. You cannot concatenate a character type to a numeric type or a date type. Each type must first be converted to a character string before they can be concatenated together.

Note

Any legitimate characters from your character set may be used in your expression as ***chr(171)****. Here is a chart of my character set.*

--=== Extended Character Set ===--

33: !	47: /	168: ¨	183: ·
34: "	123: {	169: ©	184: ¸
35: #	124: |	170: ª	185: ¹
36: $	125: }	171: «	186: º
37: %	126: ~	172: ¬	187: »
38: &	127: ▌	173: ·	188: ¼
39: '	145: '	174: ®	189: ½
40: (	161: ¡	175: ¯	190: ¾
41:)	162: ¢	176: °	191: ¿
42: *	163: £	177: ±	222: Þ
43: +	164: ¤	178: ²	223: ß
44: ,	165: ¥	179: ³	
45: -	166: ¦	181: µ	
46: .	167: §	182: ¶	

Note

Even though I have disabled Record Typing within my GoldMine the ⦿ ***Global Label, used for 'GM Dealer' record type across all contact sets:*** *still picks up my Key1 field label* ***GM Dealer****. Your terminology of this statement may vary depending on your current GoldMine Record Typing setting.*

Note

Even though you can spin the ***Data Size:*** *field value up to* ***100****, when saved, GoldMine will re-adjust this value to* ***69****.*

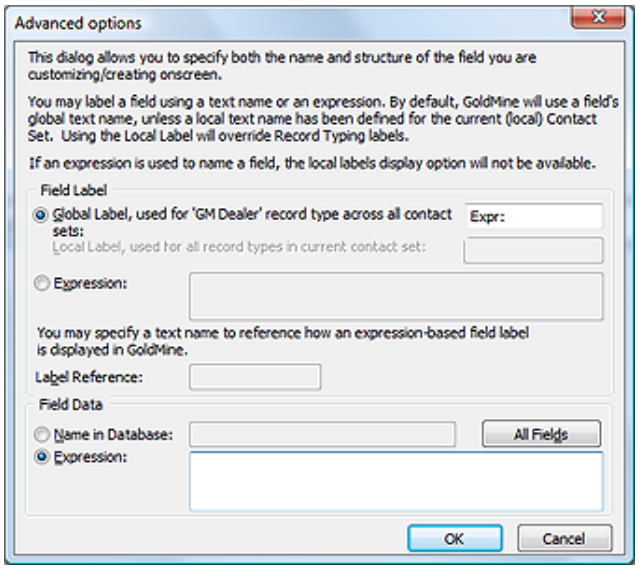


Figure 4-8

one tab **Fields**, one would want to have some sort of identification on each of the screens for the active screen if you are not using individual tabs. I will use an expression field to title this screen. I click on the toolbar **New** button, and I select **-- dBASE Expression --** from the list of fields (**Hint:** Just type a hypen (**-**) in **Field:** value of the **Place Field** dialog form). Clicking on the OK button from the **Place Field** dialog form will place the field on the screen. Double-clicking on the field on the screen, or keying the Enter key from the keyboard to bring up the Field Properties dialog shown in Figure 4-7 on the previous page.

To arrive at the **Advanced options** dialog form as of GoldMine Premium 8.5.1.12, one must click on the **More options...** button, Figure 4-7. Notice that you are positioned on the radio button ⦿ **Global Label, used for 'GM Dealer' record type across all contact sets:**, and the field contains **Expr:**. There has been a slight change here for GoldMine Premium, as GoldMine will no longer allow label names to remain blank. You must have something in the **Field Label** frame before you can leave the **Advanced options** dialog form. In order to leave the label blank, you need to trick GoldMine. I have found that if you place a colon (**:**) in this field, that this will suffice, or you can just leave the default **Expr:** value.

Next, you'll need to add your expression. This is the expression that I put into the **Field Data** frame ⦿ **Expression:** field:

```
[--===| Personal Screen for ]+trim(Contact)+iif(empty(Company), [],[ of ]+trim(Company))+ [ |===-- ]
```

Though the expression field appears to be large, and appears as though it will wrap text, it does not (go figure as my spouse would say). This expression may or may not be appropriate, but it does give you an example of what can be accomplished with an expression field. The nice part is that expressions are always evaluated in real time, therefore, as you change contact records, the expression field will change accordingly. I will give you a better example of how appropriately an expression field may be applied, when I discuss an expression field against the date of birth field later in this chapter. That expression field will be to display the age of the contact in real time.

Being an old dBase programmer, I prefer to use the square brackets (**[ ]**) when possible instead of the quotation (**" "**) marks to delimit a character string. In pseudo code (programmers use pseudo code a lot), the expression represented above concatenates a character string with a character field that has had the spaces trimmed off of it, with a function that evaluates the company field. If the company field is empty, the function adds nothing to the expression while if the company field contains information, the function concatenates a character string and the company name. Wow, that was long winded.

Let's look at the individual pieces.

[--===| Personal Screen for]	character string
trim(Contact)	trim function on character field
iif(empty(Company), [],[of]+trim(Company))	immediate if function returning character string
[|===--]	character string

The plus (**+**) sign is used to concatenate character strings together. To be correct, and to be displayed in GoldMine, the expression must result in a valid character string. For example, if we had added the age() function to the end of this expression, it would have resulted in expr error being displayed on this screen. The age() function returns a numeric value from a date field. This would have violated the rule that "The result of any expression must return a character string".

Next, we should move on to the **Layout** tab, Figure 4-9 on the next page, and we will make some adjustments here. First, in the **Field Size & Position** frame, specifically in the **Field Label Size:** field, as I don't have a label for this expression field, I will drop this value from 10 to **0**. I will then make the **Data Size:** value **69** (the maximum allowed in this field). For this exercise, I will set the **Colon (:) Row (y):** value to **15**, and I will then set the **Column (x):** value to **2**. I can now click on the **OK**

button to accept this expression, and its layout. Based on the contact record that is currently active in my GoldMine Premium, the expression immediately displays:

--===| Personal Screen for DJ Hunt of Computerese Inc. |===--

You should have noticed that the expression is evaluated immediately upon saving the expression even though you remain in the design mode. This is called instant gratification as well as letting you know that your expression is correctly syntaxed.

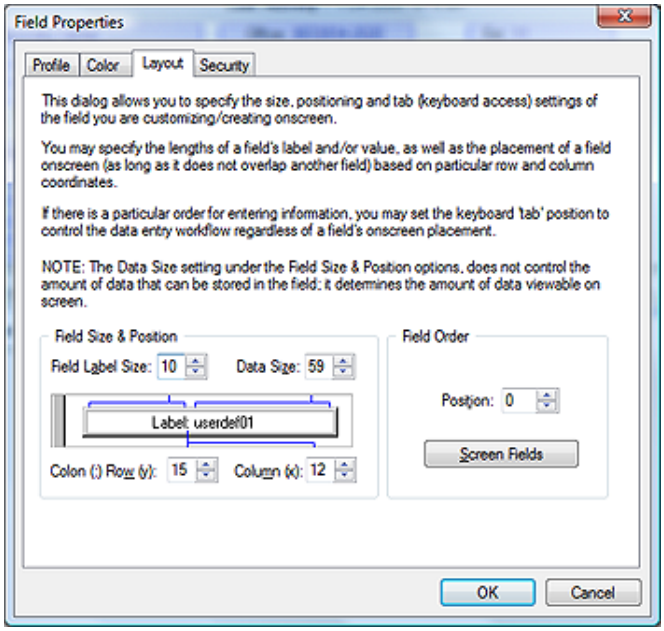


Figure 4-9

Now let's look into adding a field to the screen, and I'll make use of some of the new features that allow for field level record typing. I'm going to add the field to the **Personal Screen**, but I only want the field displayed when the record type is **Buyer**, **Seller**, or **Agent**. If the record type is **Property**, then I do not want the field to be displayed. I am using the **Contact1.Key1** field for the record typing driver. How about changing the color of the label, and the color of the data when the fields are displayed to, let's say, blue for the label and magenta for **Buyer**, red for **Seller**, and green for **Agent**. I have already added a new custom field named **ucSSN**, **C**, **11** and, while I was at it, I added another field **udDOB**, **D**, **8** (we'll use this one later in this chapter).

Add the field to the screen as I did with the expression field before, only this time select the field **ucSSN** (as it appears on our list, yours may show Social Security Nu (Ucssn)), and then double-click on the field once it has been displayed on the screen. Now let's change the **Label used for 'GM Dealer' record type:** to **SSN:**, and then click upon the **More options...** button. The resulting **Advanced options** dialog form, for this exercise, is shown here in Figure 4-10. Please note that the ◉ **Global Label, used for 'GM Dealer' record type across all contact sets:** should have been populated with the field description placed in the **Label used for 'GM Dealer' record type:**. What I show you in Figure 4-10 is what I have used for a **Global Label**. I have inserted the label **SSN:** as described earlier, which is what I would want displayed as the label when the field is displayed on a screen. Click on the **OK** button, to return to the **Field Properties** dialog form.

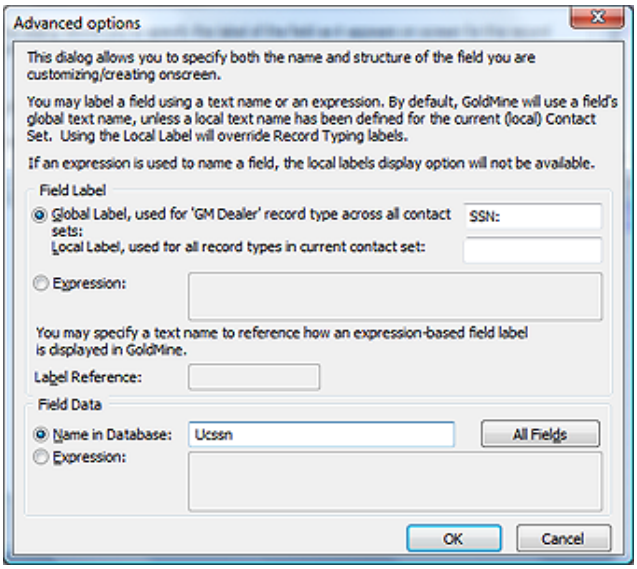


Figure 4-10

Next click on the **Color** tab, and you can see the results of this exercise in Figure 4-11 on the next page. Remember that I said that this field should show for all field level record types except **Property** (I don't know of any property that possesses a Social Security Number), and then, when the label was displayed, that it would be Blue (see sidebar **Tip** on the next page for determining the numbers to use to produce the desired colors).

In order to achieve my goal of label and data not being displayed when my record type is **Property**, and to be displayed with a **Blue** label otherwise, as well as **Magenta** for the data field on a record type of **Buyer**, **Red** for the data field on a record type of **Seller**, and **Green** for the data field on record

Note

Global Label *vrs* ***Local Label***

You might ask: "When should I use a ***Global Label*** *as opposed to a* ***Local Label****?".*

When you create a new Contact Set based on the structure of an existing Contact Set, the Global Label is carried across to the new Contact Set, however, the Local Label is not.

In theory, the Local Label is per Contact Set, while the Global Label is, well Global, and applied across all new contact sets.

Note

Social Security Numbers *usually take of the form of* ***###-##-####****. In the* ***Lookup.ini*** *chapter of this book, I explain to you how to create this presentation regardless of what your end users enter into this field.*

Another formatting option that I discuss for the Social Security Number in the Lookup.ini chapter is the ***GMTray*** *application.*

Tip

I can never remember the numbers assigned to the various colors. When I get on the ***Colors*** *tab, I immediately select the* ***Colors...*** *button. I then select the first of the colors that I want to use from the color chart, and select* ***OK****. I then write down the number that appears in the ○* ***Expression:*** *window, and I will use that number in my expression. I do this for all of the colors which I intend to be using.*

Note

*When setting colors, **-1** represents the default GoldMine color for that attribute, while **-2** hides the entity on the display. To hide a field entirely from the display, the **Label Color**, and the **Data Color** must both be set to **-2**. All other colors must be represented in the expression by a valid color number.*

Tip

When creating logical expressions, try to account for user unpredictability. For instance a user could enter into the Record Type field:

property
Property
PROPERTY
PrOpErTy

Anyway, you do want your expression to pass True for all of these cases, hence, I trim() off any trailing spaces, and convert the value to all upper() case for comparison.

trim(upper(Contact1->Key1)) == [PROPERTY]

*You may have noticed that I used the exactly equals operator (**==**) as well.*

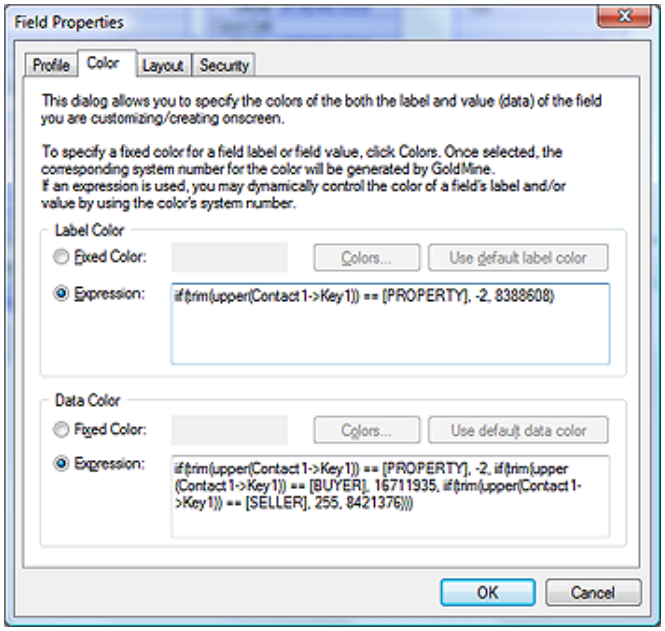


Figure 4-11

type of **Agent**, I must employ the O **Expression:** option for each. Each expression employs the **i**mmediate **if** function, iif(logical result, true, false). In fact, for the **Data Color** expression, I've embedded three levels of this function, however, let's examine the **Label Color** expression carefully first.

iif(trim(upper(Contact1->Key1)) == [PROPERTY], -2, 8388608)

My logical expression is:

trim(upper(Contact1->Key1)) == [PROPERTY]

Should that evaluate to True, I set the color to:

-2 (Hide)

Should that evaluate to False, I set the color to:

8388608 (Blue)

Are you ready for the big one? Here is my explanation of the Data Color expression:

```
iif(trim(upper(Contact1->Key1)) == [PROPERTY], -2,
iif(trim(upper(Contact1->Key1)) == [BUYER], 16711935,
iif(trim(upper(Contact1->Key1)) == [SELLER], 255, 8421376)))
```

Keep in mind that this is one continuous line of code, and is only separated here for display purposes.

My 1st logical expression is:

trim(upper(Contact1->Key1)) == [PROPERTY]

If that evaluates to True, I set the color to:
-2 (Hide)

If that evaluates to False, evaluate the 2nd logical expression:

trim(upper(Contact1->Key1)) == [BUYER]

If the 2nd logical expression evaluates to True, set the color to:

16711935 (Magenta)

If that evaluates to False, evaluate the 3rd logical expression:

trim(upper(Contact1->Key1)) == [SELLER]

If the 3rd logical expression evaluates to True, set the color to:

255 (Red)

If the 3rd logical expression evaluates to False, set the color to:

8421376 (Green)

I didn't need to test for **Agent** as, in our scenario, that was the only possible remaining record type. However, should you have more record types, then you would have evaluated for **Agent**, and on false, you might have set the color to the default color. I won't analyze it for you, but that expression could look like:

```
iif(trim(upper(Contact1->Key1)) == [PROPERTY], -2,
iif(trim(upper(Contact1->Key1)) == [BUYER], 16711935,
iif(trim(upper(Contact1->Key1)) == [SELLER], 255,
iif(trim(upper(Contact1->Key1)) == [AGENT], -1, 8421376)))
```

I haven't tested this out for you, so I can't state for certain, however, in the dBase language one is not able to embed deeper than 25 levels. In the code above, I have embedded 4 levels. You'll need to test this out for yourself if you are truly interested in seeing the depth to which you are allowed to embed iif() functions.

Let's move on to the **Layout** tab, as displayed here in Figure 4-12. Here we can see the **Field Size & Position** frame. Users tend to misunderstand this section. Any changes made here are not, in fact, making changes to the underlying data (field) structure, but are, instead, making changes to the portal that displays that information. In support of that statement, the ⦿ **Global Label, used for 'GM Dealer' record type across all contact sets:** can be as many as **15** characters long. To display that many characters on the screen one would need to increase the **Field Label Size:** from **10** to possibly as much as **15** (this is solely dependant on screen font). Changing this number to **20** would not gain one any more label space than the data field currently holds. The **Data Size:** might have been a better example. You may have a field that is 256 characters long, while the **Data Size:** should not exceed **69** (this assumes that you have a **Field Label Size:** size of **0**, see sidebar). If you are using the small fonts in GoldMine, this means that about **100** characters of your **256** character long field will be displayed. In the edit mode, you may access the rest by scrolling, however, what you are designing here is the portal to the data, and what it will display. Again, don't get the data field structure size confused with this portal display view size.

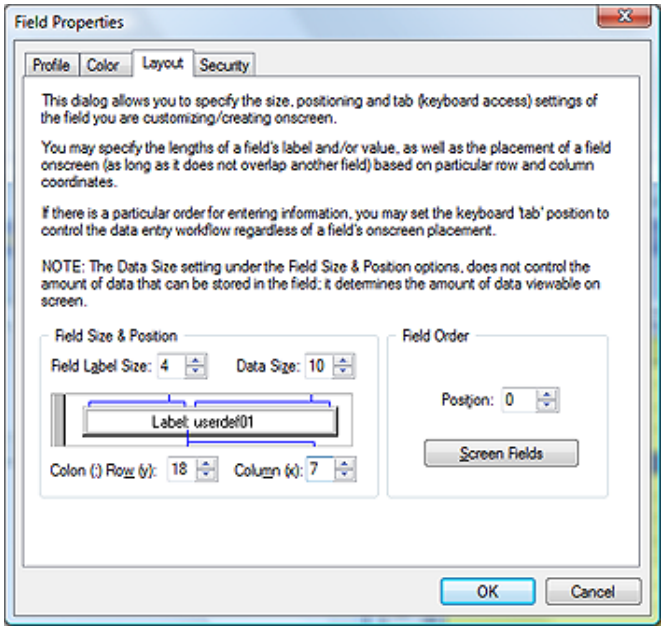


Figure 4-12

Without rehashing what I've already said a number of times, the **Field Label Size:** is where you indicate how much of the label is displayed on the screen. The **Data Size:** represents how much of the data field structure is displayed through the portal.

The bottom row of information defines where that portal is positioned on the screen. The positioning on the screen is defined based on where the **Colon (:)** would be positioned. To emphasize, the positioning is not based on the beginning of the text or the end of the text, but, instead, from where the colon between the label and the data is to be positioned on the display. Usually, on your screen, you would have a label and data, it would be displayed thusly, **Label: Data**. Though you don't always have a label (when you want to have a question followed by an answer and the question is more than 15 characters in length for example) the positioning always assumes the colon position proceeds the data, even if no label is attached.

The first **Row (y):** available under the **Fields** tab is row **15**, however, that is extremely close to the frame, and placement on this row would not be visually appealing. On the other hand, the **Column (x):** may be anything from **1** to whatever. That whatever can be a gotcha as well. You could, in theory, place the field on the screen where no one could get at it. Let's say that you made the **Column (x): 200**. You would be able to see the field move out of view into the nether world while you are in the design mode, and this is exactly what the users would see. Yes the field is there, but it is neither visible nor accessible. As a designer, you need to be cognizant of your every move. Keep that **Column (x):** position to the viewable screen position on a standard resolution monitor.

The next frame, on this tab, is the **Field Order** frame. In this frame the designer determines the tabbing order for this field as compared to the other fields in the view. The tab order should not even be available for an expression field, as the user could not possibly tab into or out of an expression field on the screen, nor edit the information contained therein. However, it is available, and I would certainly recommend that you do not even set the tab order for expression fields. If you are placing an

Note

Field Label Size: and Data Size: do not relate one-to-one with the number of characters. One may require a Field Label Size: of only 12 to display a label containing 15 characters.

Note

GoldMine will enforce its rule that the Field Label Size: plus the Data Size: may not exceed 70. Even though you can spin the number high, when saved, these will be reset to sum 70.

Note

Remember to allow for the F2 Lookup arrow that follows the data portal. I usually allow 4 places for the arrow. If I had a two character field, and I wanted to display both characters, I might increase my Data Size: to 6.

Note

If you require a Label that is larger than the 15 characters allowed by GoldMine then you are advised to utilize an Expression field for the label as I had discussed earlier in this chapter.

Tip

Personally, I wait until the screen is totally laid out, at which time I return to set the Tab Order. I feel that this saves a lot of extra steps.

editable field on the screen, however, then it is wise to set the tab order, **Position:**, for those fields. To us, the term **Position:**, is confusing to the designer. Do not get confused, setting this number is setting the **tabbing order** number, and tabbing is done in sequence. The spinner control to the right of the field will increment the number from **0** to **100** in increments of **1**. As you could have up to **250** fields per screen, if you wanted to change the tab order for all 250 fields, you would be required to type the tab order by hand beyond the 100 spinner number limitation.

Tab Order - When one has placed a field on the user defined screen in edit mode, and then depresses the **Tab** key on their keyboard, the cursor will advance to the next field in the sequential tab order.

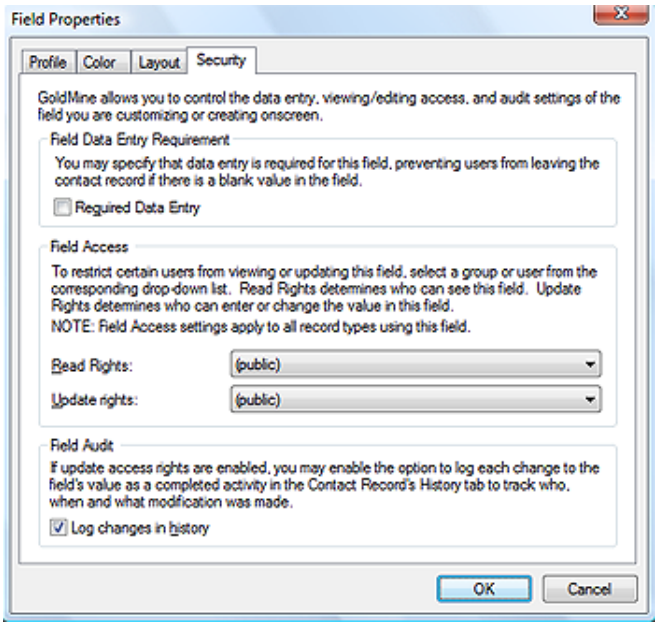


Figure 4-13

We can now continue on to the next tab, that being the **Security** tab, Figure 4-13. The first frame, **Field Data Entry Requirement**, contains but one option, ☐ **Required Data Entry**. As the dialog states: *"You may specify that data entry is required for this field, preventing users from leaving the contact record if there is a blank value in the field."* I believe that defines this satisfactorily.

The first selection, in the **Field Access** frame, is the **Read Rights:** field, and then the **Update rights:** field. This is the same type of security that we had access to when we were setting up the screens, however, this access is applied at the field level as opposed to the screen level. If you wanted a certain user or group of users to be able to see this field, then you would assign that user or group of users to the **Read Rights:** field. The same applies to the **Update rights:** field. A typical case may be that you want some users to be able to see a field, but you might not want those same users to have the ability to update the information in that field. The default for each field is **(public)**, which, of course, means everyone has the ability to see, and to update the data value in the field. Remember, in either of these cases, you are assigning the user(s) that will have **Read Rights:** and **Update rights:**. Field Access rights are a very good reason for defining your user groups (discussed earlier in Chapter 2) appropriately, and remember that users may belong to more than one user group. Take your time when designing your user groups. You can always modify them later, but make them as useful as possible when you begin your design. As this is the Social Security Number field, you may want only certain individuals to enter this information into the field (**Update rights:**).

Note

*You may remember that I discussed the UserID **Access** property **Required field override** option back in Chapter 2. Combined with this option, the option discussed earlier may now supply you with a better understanding of Security policies within GoldMine.*

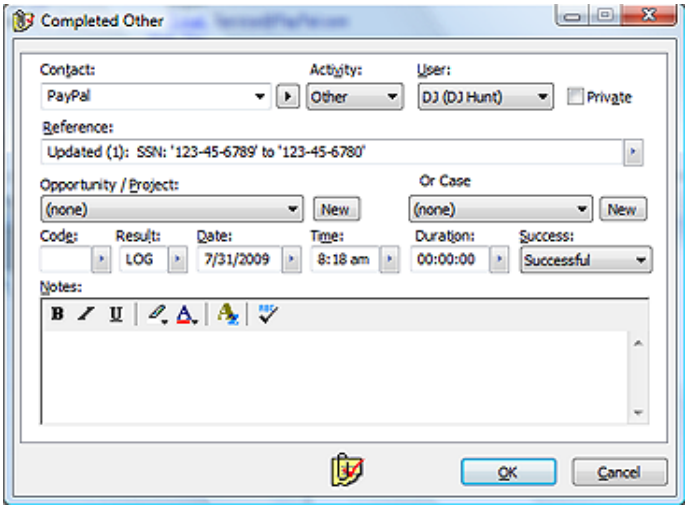


Figure 4-14

The next frame, on the **Security** tab, is the Field Audit frame which, again, contains a single option. This option is the ☐ **Log changes in history** option which is not checked in its default state, however, for the SSN I have chosen to select it. It would have been good practice on the part of FrontRange to have disabled this option when creating an expression field. By their very nature, an expression field cannot be changed. However, as this is a field that I am placing on my screen, I may want to check this

Note

*In past versions of GoldMine, selecting the ☐ **Log changes in history** option, would not have created a History record had the field been blank prior to the field change.*

As of GoldMine Premium 8.5.1.12, FrontRange has rectified this. Whether the field is blank or contains a value, when modified, a GoldMine History activityis created.

option. The Social Security Number is a unique number, and shouldn't change ever, once entered. I would want to know what, when and who changed this field if it were ever to change. Any change in the field being monitored would result in the creation of a History Activity under that contact record with all of the pertinent information.

Figure 4-14, previous page, is the result of a change made to the **SSN:** field that had been set to ☐ **Log changes in history**. GoldMine shows you the userID that changed the field, as well as the date and time that the field was changed. Notice also, that GoldMine maintains the old value as well as the new value of the field being monitored for change in the History **Reference:** field.

```
Updated (1):  SSN: '123-45-6789' to '123-45-6780'
```

Selecting this option for more than a couple of critical fields could result in a fast growing **ContHist** table, and therefore, I do not recommend that you abuse this field monitoring convenience especially if your remote users are working against the SQL Server Express 2005/2008 backend with their database size limitations. Select this option for your most mission critical fields, and only for as long as monitoring is deemed necessary. With the proper security settings, monitoring a field for changes should not even be required unless you suspect that there is a problem within your company.

Earlier, I mentioned that I would add a field on the **Personal** screen, **udDOB**. If you are doing the same, I would like you to set the Color attributes exactly as I had done for **ucSSN**. You see, I would not want this field to be displayed if the **Record Type** were that of **Property**. Now if one assumes that this is the persons **Date of Birth**, one may want to display the individuals age on this screen so that a user reviewing the information will not have to pause to calculate the individuals age while in the middle of a conversation with that individual. To do this, I would ask that you add an expression field. This is the expression field that I employed:

```
iif(dtos(Contact2.uDOB) > [], [Age: ]+ltrim(str(age(Contact2.uDOB))), [Age: Date of Birth Empty])
```

Remember to put a colon (**:**) into the **Label used for 'default' record type:** field. Remember to set the **Color** attributes identical to those of **ucSSN**. I would not want this age field to display for a **Property** record type, though I might create an alternative to display, showing the age of the property.

As a quick source:

Label Color

```
iif(trim(upper(Contact1.Key1)) == [PROPERTY], -2, 8388608)
```

Data Color

```
if(trim(upper(Contact1.Key1)) == [PROPERTY], -2,
iif(trim(upper(Contact1.Key1)) == [BUYER], 16711935,
iif(trim(upper(Contact1.Key1)) == [SELLER], 255, 8421376)))
```

Lastly, remember to set the **Field Label Size:** from **10** to **0** as we are generating our own label, **Age:**.

Now, whenever the user is on a record, of **Record Type: Buyer**, **Seller**, or **Agent**, and there is a value of **11/23/1948** in the **udDOB** field, the age will be displayed as:

```
Age: 60 ( at least it was on July 31st, 2009 )
```

With no value in the **udDOB** field, the age will be displayed as:

```
Age: Date of Birth Empty
```

Here a little tidbit that I discovered in the forum. To change the color of a single word based on the 1st character of the word, you could try this expression:

```
color(word('Black Blue Red Green Yellow Purple Orange Gray', at(left(Contact1.Key1,1), 'ABCDE-
FGN')))
```

Notice that I am applying this based on the 1st character in the Key1 field. I would also like to mention that, although it does work, it does not work as expected in GoldMine Premium. The entire expression value is converted to the color based upon the 1st letter in the Key1 field.

Record Typing

What exactly is **Record Typing**?

Well, I have been discussing **Record Typing** of sorts in this chapter already without specifically pointing a finger to it. I have been discussing **Field Level Record Typing**, and this was my preferred method of **Record Typing** within GoldMine. There is, however, **Record Level Record Typing** inherent within GoldMine Premium which has improved tremendously since its inception in the GoldMine

product. In its basic terms, the GoldMine Administrator would designate one GoldMine field as the **Record Type** field. I'll work with record typing as designed for use in an automobile dealership as my subject matter for this explanation (this is the sample configuration GoldMine Premium). Therefore, if I were to use, let's say, **Contact1.Key1** as the **Record Type:** field, I would then create an accompanying **F2 Lookup List**, refer to Chapter 6, that contained **Buyer**, **Service**, **Agent**, and **Vehicle**. Others could be added to this list, however, this should suffice for this section of the chapter. I would then want to lock down that list to not ☐ **Allow blank input**, and to ☑ **Force valid input**, while it might be nice to give a **Field Name:** for this F2 Lookup List of, oh let's say, **Record Type**. I might also want to check the option to ☑ **Auto Fill**. This field would then be employed to designate what type of information a particular record is going to contain. For example, if the **Record Type:** field contained **Buyer** on a particular record, then one would know that all of the information contained on this record, as well as the screen design, would belong to the **Buyer** record type. Whereas, if this field were to contain **Vehicle** on a particular record, then one would know that the screen design and fields would represent the vehicle on the lot to be sold or delivered. By the way, in this scenerio, the Relationship Tree is your best friend. You could relate an Agent, to a Buyer, to a Vehicle for easier visualization. How sweet (hmmm - efficient) is that?

All other GoldMine **Contact1** and **Contact2** table field **Labels**, **Color** and **Display** may be based upon this one field. I have discussed **Field Level Record Typing** in more detail in the **Screen Design** section of this chapter, however, it is important that one understand the concept now. As well, it is important to understand that the combination of **Field Level & Record Level Record Typing** is highly recommended for the best possible presentation of **Record Typing**. Additionally, which view displays in default under **GM+Views** may also be based upon the type field. Therefore, based on this one field, the end user could see pictures of the vehicles, if the record were designated a **Vehicle** type, or the end user could see the buyers interest and history if this were a record type of **Buyer**.

Let's take a look at the default **Contact1.Company** field. This field is a **Character** based field having a field length of **40** characters (refer to Chapter 9). No matter what one does with the display of this field, the underlying characteristics of this field will never, ever be changed. That is an important concept to accept in your understanding of record typing. When looking at the GoldMine display, and when having selected either **Buyer**, **Service**, **Agent** or **Vehicle** in the **Contact1.Key1** field, it might be appropriate to have this field label display **Buyer:**, however, if the **Contact1.Key1** field were to contain **Vehicle**, then it might be more appropriate to have this label now display **Vehicle:**. Mind you, the underlying field will not have changed, it is still **Contact1.Company**, **Character**, **40**, however, it would be presented to the end users as **Buyer:** or **Vehicle:** depending upon the value in the **Contact1.Key1** field. As well, I could easily change the color of the **Label** or **Data** in the display based on the value in the **Record Type:** field.

So, what is **Record Typing**?

Well, Record Typing is nothing more than the ability to use one table for different types of data input. FrontRange has been using this concept for years without the end user being fully aware of this. Yes, the **ContSupp** table employs a field called **RecType** that determines what type of GoldMine information is contained in any given **ContSupp** record. You'll read about this later in Chapter 9, however, let me show you this ahead of time:

■ The following are possible values for the **ContSupp.RecType** field:

A	**Record Alerts**	**L**	**Linked document**
C	**Additional contact record**	**O**	**Organizational chart**
E	**Automated process attached event**	**P**	**Profile record/extended profile record**
H	**Extended profile header**	**R**	**Referral record**

So one can easily see that GoldMine is using the **ContSupp** table to store **Record Alerts**, **Addition Contact Records**, **AP's**, etc., and all without change to the underlying table structure. GoldMine takes this hard coded record typing to the user level, and let's you apply this same concept against the **Contact1**/**Contact2** tables as well as **GM+Views**. You may designate one or more fields to define your Record Type, and based on the Record Type the, let's say, Company field could contain a company name, or a part number, or anything that you design dictates. Very neat customizability indeed.

Up until this book, that which was previously stated was the extent of the Record Typing section of this chapter, and, although I still feel that it has issues, I will expand upon Record Typing for the GoldMine Premium edition of this book. That's right, here and now.

One must be a user possessing Master Rights in order to select

Tools
Configure ►
Record Types

from the GoldMine menu which, by doing so, will return a dialog form similar to that which I am showing below in Figure 4-15. I have expanded all of the trees so that you will have a clearer picture of the **Record Type Administration Center** as it has been distributed with the GoldMine Premium Edition.

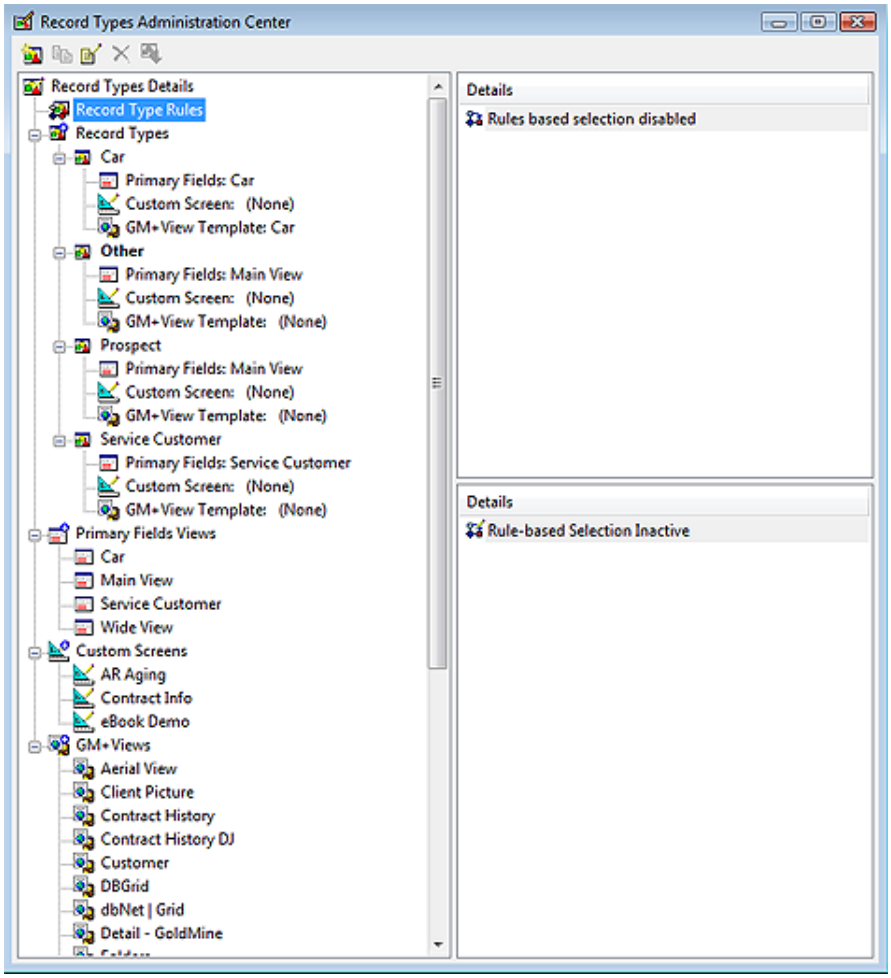


Figure 4-15

To achieve the most that you can out of the Record Type Administration Center capabilities, you want to work backwards, in effect, climbing your way up the tree. In fact, prior to even entering the Record Type Administration Center, you would want to create any and all of your potential GM+Views, and Custom Screens. The Custom Screens are only necessary if each screen is unique. If, however, you plan on utilizing one screen for different Record Types (recycling fields) then you would use the Clone capability of the Record Type Administration Center. Wow! I think that I confused myself there. Simply put, unique screens are developed with GoldMines Screens Setup... utility while re-used screens will be designed via cloning in the Record Type Administration Center.

Under the **Custom Screens**, shown above in Figure 4-15, you will see that I have already created 3 distinct custom screens: **AR Aging**; **Contract Info**; **eBook Demo**. Let's begin by cloning the **Contract Info** screen. This is a screen that I had created earlier for our companies actual GoldMine usage. Right-click on the Contract Info screen, and select **Clone** from the local menu. Your resulting dialog form should be an old friend, the **Custom Screen Profile** dialog form. Notice in Figure 4-16, that I have added a screen name of **Record Typing Sample**, and a tab name of **Sample** for this exercise. Clicking on the **OK** button will then add the **Record Typing Sample** screen to the **Custom Screen** tree, Figure

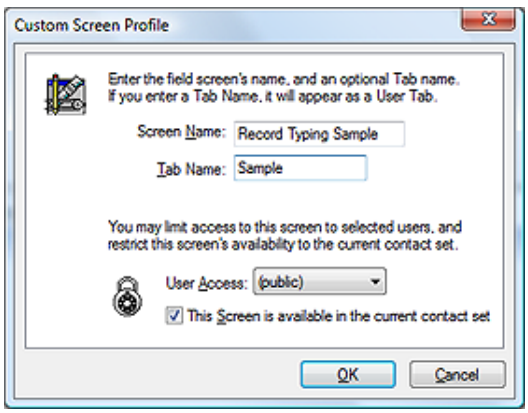


Figure 4-16

Note

*You cannot add fields to a screen via the **Record Types Administration Center**, so plan ahead. Begin with a User Defined screen that has many fields on it, and then **Clone** it for your new screen with reusable fields. Field reuse is the key for Record Type screens.*

WARNING

Issue:

Using Local Labels in Record Typing

***Response:** (Grant Ellsworth/2008)*

I found evidence that Local Labels are not associated with the specific Record Types.

Here's how/why:

The Record Types, and the associated field labels or Global labels, are stored in the Fields5 table (refer to The Tables chapter); Local Labels are stored in the database's ContUDef table, making the Local Label global to all record types. Hence, I use only Global Labels for the Record Type labels. This consistently works.

Hence, Local Labels and Record Typing are a non-intersect set, an oxymoron, if you will.

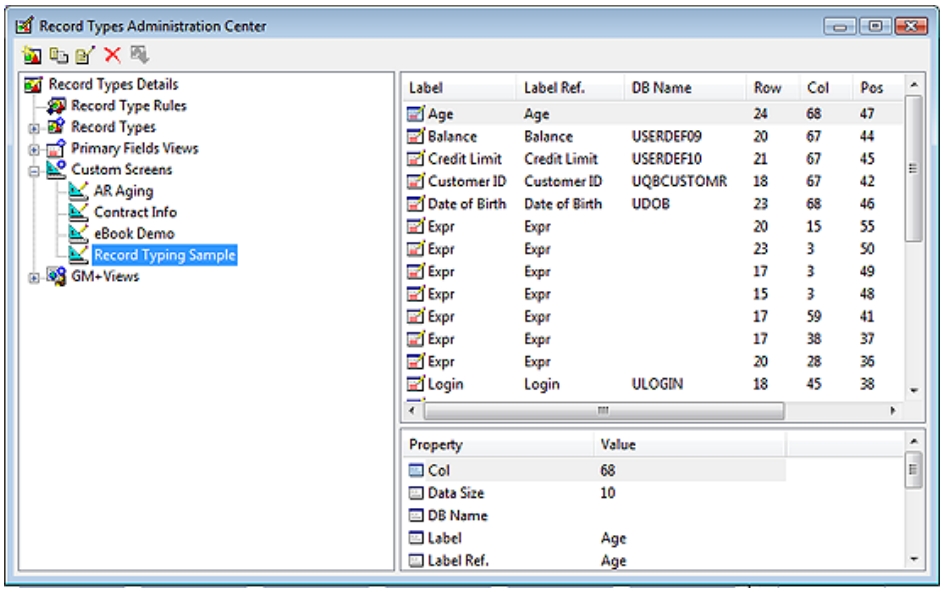


Figure 4-17

4-17. If the **Record Typing Sample** is not already highlighted, please highlight it now. If you look in the upper right hand side of the dialog form, you should see all of the fields/expressions that we had previously added to the **Contract Info** screen repeated exactly on this screen. Below that, if a field is highlighted, you will see the **Properties** associated with the highlighted field.

If you right-click on, let's say the **Tilde (~) Usage** field (not shown in Figure 4-17) or any field for that matter, and then you select **Edit...** from the local menu. Next, if you select the **More Options...**, you would bring into focus, our old friend, the **Advanced options** dialog form. To understand this, you must position yourself properly within your mindset. You are editing some of the field properties for this field only as it relates to the **Record Typing Sample** screen. You will, for instance, be able to change **Field Label** as I have done here in Figure 4-18. You will also be able to change the **Field Data**, so although you cannot add a new field to this screen, you will have the ability to change the data field to another predefined field or expression. You will have full access to all properties on the **Color** tab via the **Field Properties** dialog form. You will not have the ability to change any of the properties on the **Layout** tab except for the **Field Order**. You will have full access to all properties on the **Security** tab. Let me reiterate, however, that any changes that you make via the **Field Properties** or the **Advanced options** dialog form here, are only relevant to that field when the end user is on the **Record Typing Sample** screen. You have to always remember and think of this as field recycling.

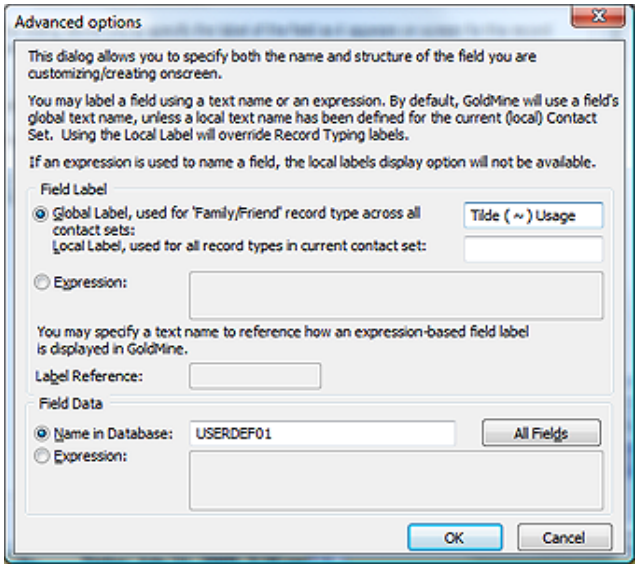


Figure 4-18

Let's climb up that tree a bit further now to the **Primary Fields Views** branch. GoldMine Premium comes standard with four primary views: **Car**, **Main View**, **Service Customer** & **Wide View**. You may remove any of these, if they are not necessary in your schema, by right-clicking on the view, and choosing **Delete** from the local menu. You may want to keep these here as examples. The choice is strictly yours. I just had to figure a way of discussing the ability to delete views that exists in the **Record Types Administration Center**, and to that end I have succeeded.

Now let's say that you had your own record type, and that you wanted to reutilize a screen to define your own **Primary View** for that particular record type. Again, right-click on the view that you wish to use as your base, and select **Clone** from the local menu. This is nothing different than that which

we just accomplished for the new **Custom Screens**.

Shimmy on up that tree another branch to the **Record Types** branch. Let's highlight that branch by selecting it, and then right-click and select **New Record Type** from the local menu. The resulting dialog form is shown here in Figure 4-19. It is here that we can bundle our **Record Types** into a package that could include one leaf from each of the underlying branches.

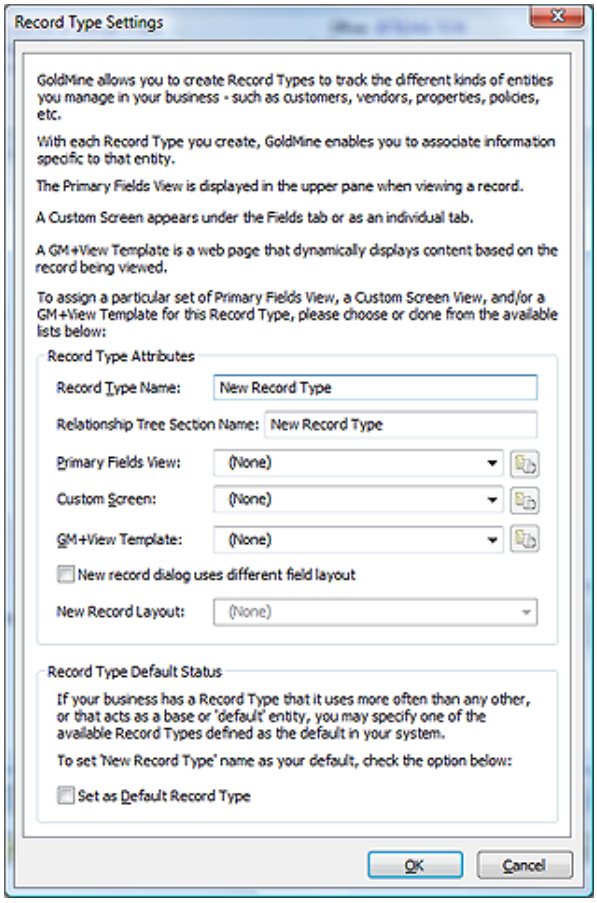


Figure 4-19

In the first frame, **Record Type Attributes**, we have to make our choice of a view for each category, however, first you would want to choose a **Record Type Name:** for your bundle. As an example only, I will type **Sample Record Type** into this field. You feel free to do as your creative mind instructs you to do if you are following along with this example that is. Now we must instruct GoldMine as to what screens to utilize as the default screens for this record type. The first screen necessary would be the **Primary Fields View:** screen, then on to the **Custom Screen:**, and, finally, the **GM+View Template:** screen. In all cases, you will only be able to select a single screen from your already defined screens contained in each branch.

Note

*Notice that I said the **default screens** for the Record Type. Obviously the user could select from any of the available **Custom Screens** or **GM+Views**, hence, I am talking about the default state when a user sets focus to a record of that particular record type.*

WARNING

*You are strongly advised to **not click** on the pages icon at the end of the **GM+View Template:** field if you are utilizing GoldMine Premium 8.5.1.12. In no way does it function as expected, and it will only add more GM+Views with the same name yet with an incremented number as:*

WebSite
WebSite (1)
WebSite (2)

Let's use the **Primary Fields View:** for this example. Once you have select a view to utilize for this record type, you may further modify its properties by clicking on the pages icon to the right of the view name which, in turn, will pop this **Main View Profile** dialog form shown here in Figure 4-20. The dialog form title will vary depending on the view that you are setting such as the **Custom Screen Profile**. It is from this dialog form that you may set a **Name:** for the main view or custom view, depending upon which dialog form you are in, and you may also limit the access to this view to a single UserID or a GoldMine User Group via the **User Access:** drop list.

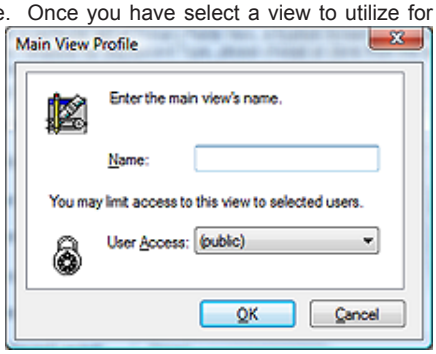


Figure 4-20

Returning to Figure 4-19, you will notice one more option in the **Record Type Attributes** frame, that being ☐ **New record dialog uses different field layout**. Although grammatically lacking, what this option is saying, when selected, is that one would like to use a different **Primary View** when they are creating a new record for this record type. If you do select this option, then you will be required to select the alternative **Primary View** to be utilized when creating a new record of this type.

There is one more frame that we have to consider in this dialog form that being the **Record Type Default Status** frame. You may only select this option for one record type in your GoldMine. As it clearly states on the dialog form: *"If your business has a Record Type that it uses more often than any other, or that acts as a base or 'default' entity, you may specify one of the available Record Types defined as the default in your system."* If this is to be that **Record Type**, then you would simply check the ☐ **Set as Default Record Type** option. Now, by clicking on that **OK** button, you will have defined your own **Record Type**. You may now continue on and design as many record types as you will require for your organization.

But wait, we're not done yet. GoldMine doesn't, as yet, know when to apply these **Record Types**. Climb up that tree just one more branch now, will ya? Right-click on the **Record Type Rules** branch, and select **Edit...** from the local menu to bring up the dialog form shown, next page, in Figure 4-21.

Tip

It is a good practice to disable your record typing rules while designing the various record types.

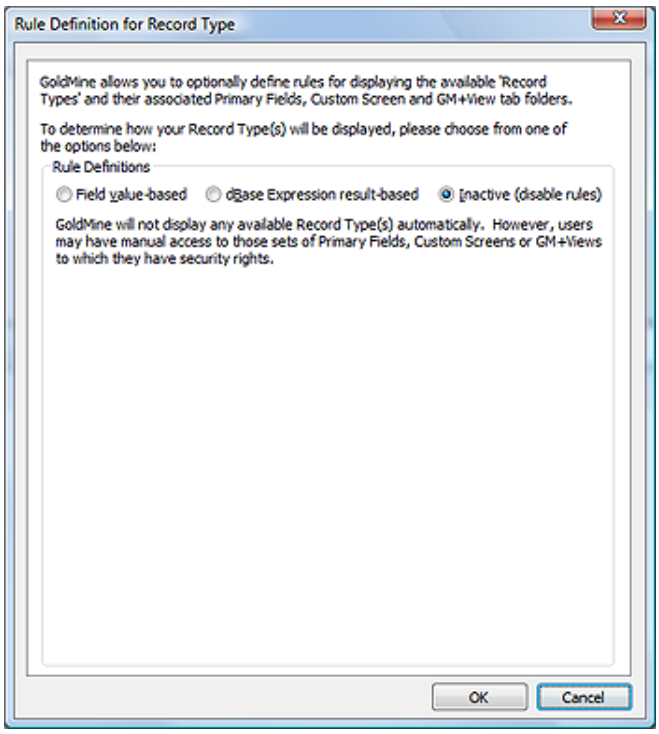


Figure 4-21

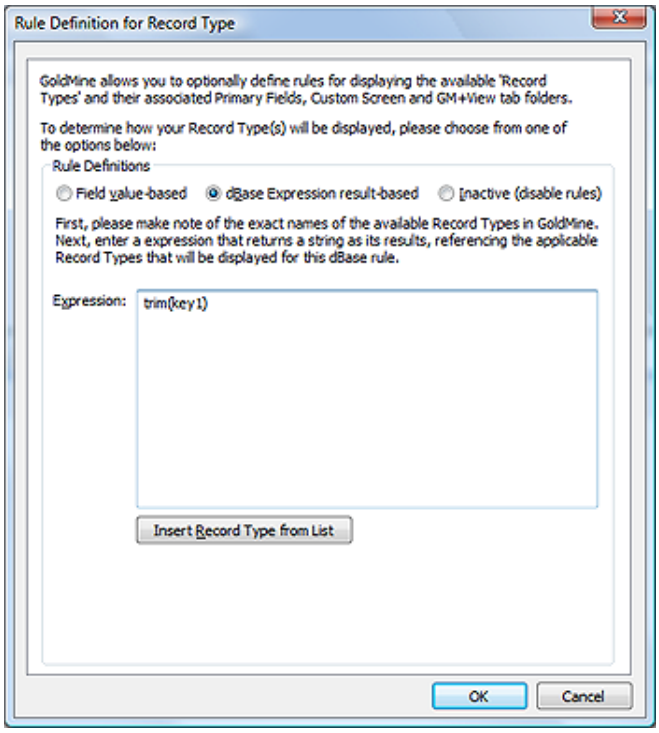


Figure 4-22

It is here that GoldMine allows you to define the Rules by which the different record types are displayed with one of the possibilities being to make the rule ○ **Inactive (disable rules)**, however, doing so would leave me with nothing further to discuss. Oh, and by the way, Figure 4-21 does not represent the default state of this dialog form. Read the sidebar Tip, and I think that you'll understand why it was in this state for my screenshot. Let's bypass that for the time being.

You may have noticed that the radio button ◉ **dBase Expression result-based** is selected in the default configuration in your GoldMIne, and that the developers have included the **Expression: trim(Key1)**, refer to Figure 4-22. I would like to mention that this is really not a very relevant expression for this example. I think that I would have chosen one that included the use of the iif() function like:

```
iif(upper(trim(Contact1->Key1))==[CAR], [Car],
iif(upper(trim(Contact1->Key1))==[PROSPECT], [Prospect], iif(upper(trim(Contact1->Key1))==[SERVICE CENTER], [Service Center], [Other])))
```

However, before I went through all of that trouble, I think that I would just have selected the first option: ○ **Field value-based** which is more appropriate for the type of iif() function that I utilized. I would suggest that the dBase expression is best used for compound iif() expressions. Figure 4-23, on the next page, is the dialog form that is the result of having selected the ○ **Field value-based** option. You can see that when you don't require compound expressions, and the record type is based on a single field value that this is probably the best approach.

You'll notice that the Reule Definition for Record Type diloag form is a little discontinuous here. You get the statement: *"First, please choose the field to evaluate in this rule. Next, choose the value required for the rule to be applied."*, but then a set of radio buttons are thrown in between this statement and the **Field Name:** selection option. Well this is all new as of GoldMine Premium 8.5.1.12, and the developers just choose to toss it on the dialog form willy nilly. So let's continue on with this section first.

WARNING

*I would warn you that if you are making any changes through the **Record Types Administration Center** that you should have everyone close GoldMine, and then restart GoldMine after you have made your changes.*

Record type for new company/contact: is the leader for this radio button selection group, and your choices are:

- ○ **Use a default record type when creating new records**
- ◉ **When creating a new record, automatically use the current record type in view**
- ○ **Set record type manually when creating new records**

As always, the Help files are lacking and fail to help us understand these options so I'll just give you my best interpretation of what I am reading.

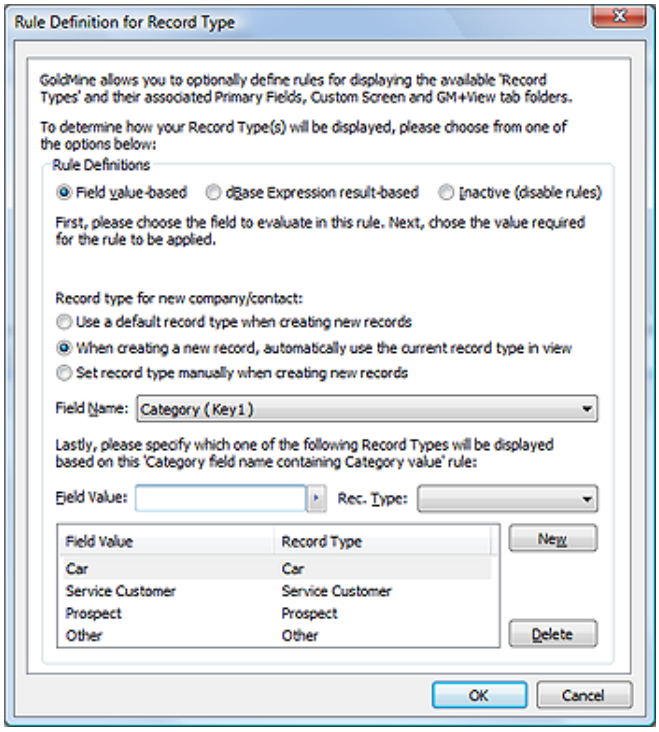


Figure 4-23

First, ○ **Use a default record type when creating new records** actual means that GoldMine will use **the** default record type, as defined by you earlier, as the record type for all newly created records.

Your second option, ◉ **When creating a new record, automatically use the current record type in view** is actual functioning as expected. What ever the record type for the active GoldMine Contact record is, when you create a New Contact record, will be the record type of the new contact record.

And, lastly, ○ **Set record type manually when creating new records** permits you to select the record type via the **New Record** dialog form **Record Type:** drop list which defaults to **[Plain Contact Record]**. As I see it, this is unnecessary as the GoldMine **File** | **New** ► menu offers one option for creating records of your various record types once you have selected to use ○ **Field value-based** for your rule. In the default state my GoldMine menu shows:

New Record

New Default Record Type...
New Car Type...
New Other Record...
New Prospect Record...
New Service Center Record...

Well then, now we appear to be back on track with the order of this dialog form so let's refresh ourselves on that previous statement again: *"First, please choose the field to evaluate in this rule. Next, choose the value required for the rule to be applied."*. You must first choose the field that will contain the value for the rule to be evaluated itself against. Since there is a finite set of fields the **Field Name:** value is selected from that finite list, and you are not permitted to input this information directly by hand. After you have selected the field name, you may add as many possible **Field Value:/Rec. Type:** value pairs as you wish by selecting the **New** button. You will notice in the default GoldMine configuration that the developers have included four possible value pairs.

This pretty much concludes the **Record Typing** section of this chapter, and for that matter, it pretty much wraps up the entire chapter as well. I would like to state, again for the record, that in the past the **Record Level Record Typing** feature of GoldMine has not been the most predictable feature with GoldMine. With each new version & build of GoldMine it has matured impressively. If working properly, **Record Typing** could enhance your database usage tremendously. Personally, I still utilize **Field Level Record Typing** for all of my needs as my industry does not require the reuse of fields in tables. It is true that **Field Level Record Typing** takes a lot more work, however, I have found it to be more stable, in the past at least, than **Record Level Record Typing**. I'd be most interested in hearing your opinions on this.

5

In This Chapter

GoldMine Premium Edition 8.5s release is causing me to have to rewrite the entire F2 Lookup List chapter for the first time as FrontRange has finally added the often requested **Referential Lookup List**. I will still cover the basics as I have in the past, however, you can see in the sidebar that I have also added the new Referential Lookup List. I, as are we all, am extremely excited to finally see the inclusion of the Referential Lookup List into the GoldMine product line.

Consistency is the name of the game when it comes to database systems of any kind. I'm sure you will remember the acronym **GIGO** for **G**arbage **I**n, **G**arbage **O**ut. This is true in all database systems where one would require report output from the data entered into the database. It would be rather difficult to filter a database on a field unless that field contained a modicum of consistency that one could apply a filter against. Is it possible that you could know all of the possible variants for a piece of data such that you could even conceive of a possible filter condition to extract that data for exports or reports? I think not.

One of your first tasks should be to establish a vehicle which will provide for as much consistency of data which is entered into the GoldMine fields as is possible. GoldMine provides, for each and every field in its database an F2 Lookup List from which the pop-up list draws its choices. The name F2 Lookup List is a carry over from older versions, although it still functions today, such that, when the field is in the edit mode, the user simply hits the **F2** key, to pop-up a **Field Lookup** list of possible choices for that particular field. Having a well developed lookup list system in your GoldMine, could eliminate as much as 90% of the end user typing that would usually be required in a Customer Relationship Management system such as GoldMine. When you combine that with the Lookup.ini instruction set that I will be discussing later in this book, well, you can have a very clean and consistent data entry system.

I recommend that you establish just such a system, and that you further develop a common process methodology that advocates the usage of the F2 Lookup List whenever possible. To this end, you will find **Auto-Fill** to be a valuable friend, however, I will cover that later in this chapter. I just wanted to plant the seed in your mind early on in the chapter.

The Basics

GoldMine, out of the box, is set up for use by FrontRange Solutions as a solution for their own internal needs. All of the field lookups relate to GoldMine in one manner or another, and as a GoldMine Administrator, it is your responsibility to make your GoldMine function efficiently for your organization.

Tip

Think through all of the possible choices for each and every field in your GoldMine installation, and add those to your F2 Lookup List. However, remember that a lookup list that has too many choices will probably be used as much as one that has no choices. There is a delicate balance that you must achieve between no list at all, and too large a list.

Let's think about when it would be appropriate to supply a lookup list for a field, and when one should not be supplied for a field. Some fields are to contain unique values each and every time data is entered into these fields. Some of these types of fields could be the **Company**, **Contact**, **Address1**, **Address2**, and **Address3** fields. Each of these fields requires unique information each and every time that data is entered into the field 98% of the time for most organizations. Some might argue that the **Company** field could contain repetitive information, and one should, therefore, create a list for those company names that are repeated frequently. I might agree with that argument under certain constraints only. If your organization dealt with a very limited number of companies, and each of these companies contained multiple sites, then, yes, I might see a requirement for a lookup list associated with this field. However, few organizations deal with such a finite number of large companies that would have multiple sites, at least in my observations.

The key word, in the previous paragraph, is repetitive. If a field can contain information that would be entered repetitively on different records, then you, most probably, have a good candidate for a lookup list entry. Such a field might be the **Source** field. Let's look at the **Source** field as our example. To display just the lookup list, while the field is in the edit mode, just click on the arrow (►) button to the right of the field. From the edit mode there are only

Note

*After 20 years of consistency in this area, FrontRange has finally decided to follow the **Windows Standards**. No longer can you **Right-Click** when a field is in the **Edit Mode** and expect to bring up the **F2 Lookup List**.*

*This is going to take some getting used to for us GoldMine old-timers. Even after eight months of using GoldMine Premium 8.5, I still have a tendency of right-clicking in the field. After a **Stupid Shit** mumbled comment about myself, I then go ahead and click on the arrow (►) icon at the end of the field.*

Trust me, eventually you will get the hang of it.

Note

*Items in a lookup list can only be removed one at a time. If you have a substantial list of entries, you may want to consider using a 3rd party product like **GoldBox** which allows you to delete entire lists in a **Sync Aware** manner.*

*Alternatively, you may want to delete this list en masse through the **SQL Server Management Tools**, but be forewarned, doing so in this manner will cause your deletions to be **Sync Unaware**.*

Note

*You could use the **Field Name:** field as a mini instruction set to your users. As this is independent of the GoldMine field name, it is pretty much free form.*

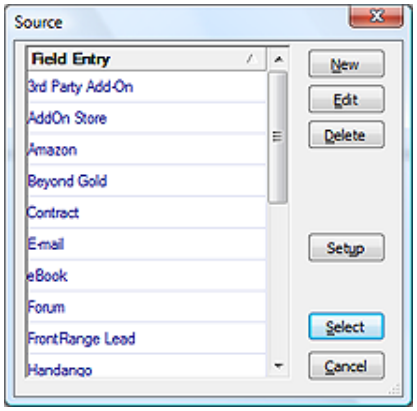


Figure 5-1

two ways in GoldMine Premium 8.5 that you can enter the F2 Lookup List window once the field has been placed into the edit mode. The first is to simply click on the **F2** key. The second method would be click the ► icon to the right of each field that has been placed into the edit mode. Either of these methods would bring up the dialog window shown here in Figure 5-1.

As you can see, there are a few entries in my Source F2 Lookup list, however, I only want to examine the buttons that are available to us at this time As you will have your own Source item list. I would add that these items are only meaningful to Computerese Incorporated, and that they are only here as examples as to how this lookup list may be utilized. In other words, you may delete all of the default entries as they are probably meaningless to your organizational needs.

It is important for you to understand that all of the buttons shown in Figure 5-1 pertain to the lookup list dialog form only. This is a very important distinction, and one that you must come to understand. Clicking on the **New** button will allow you to add a new entry to your lookup list. There are a couple of points to remember here, the first being that this entry will hold up to 40 characters. In this lookup window, however, only about 28 characters of the possible 40 will be displayed.

Clicking on the **Delete** button will simply delete the selected lookup list entry from the list. It will not delete any information from the Contact record itself. You will receive the standard warning message: *"<I<UserID>>, Delete the selected F2 record?"* to which you must select **Yes** if you really intend to delete this entry, or **No** if you did not mean to delete the selected entry.

Next we come to the **Edit** button. Clicking on this button will allow you to edit the selected lookup list entry. This is just in case an entry should have gotten onto the lookup list that is misspelled or was entered by someone else, and that may not conform to your corporate standard. This just eliminates the need to delete the entry, and then re-add it as a new entry.

I'll discuss the **Setup** button in a minute, but first let's look at the **Select** button which simply selects the highlighted entry, and enters it into the field currently being edited on the Contact record. You may bypass the need to use the **Select** button by just double-clicking on the appropriate entry from within the lookup list. The selected lookup list entry will then be sent directly to the field that is currently being edited.

Next we have the Windows obligatory **Cancel** button which, when clicked upon, will close the lookup list dialog without entering any value from the list into the field being edited.

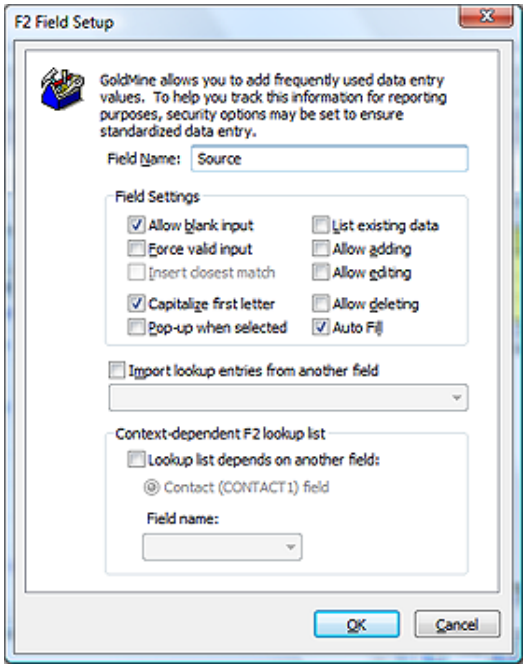


Figure 5-2

Now let's step back, and examine that **Setup** button. Clicking on the **Setup** button will bring forth the **F2 Field Setup** dialog screen shown here in Figure 5-2. The first item that we see in this dialog screen is the **Field Name:** field. Please don't confuse this with the field name that is being displayed on the GoldMine screen. The name that you enter into this field is the name that is displayed in the title bar of the lookup list, as shown, previously, in Figure 5-1. In that case, the name **Source** was the entry as it was the name that appeared in the title bar of Figure 5-1. It is always a good practice to have the name of the lookup list match that of the field to which the lookup list is associated. This way the user will always know which field they are editing. There is an exception to this rule.

Exception: In those cases where you are employing Record Typing for a particular field, in which case, you may want to leave this field blank.

Now we can look at the frame labeled **Field Settings**. ☑ **Allow blank input** is the first

Note

*You may remember, in Chapter 4, when I discussed **Form Level Validation** which allows you to set a field(s) for required input. In addition to that, once they have placed the field into the edit mode, you may then prevent them from leaving said field if they have not entered information and/or appropriate information into that field.*

Tip

*If you uncheck the box for ☑ **Allow blank input**, and check the box for ☐ **Force valid input**, you must make certain that there is an out for the user. An out might be the use of **Other**, or **Miscellaneous**. Do not leave the user stuck in the field because there is no entry on your list that meets their particular condition. If you don't supply an out, then the user, in this case, would have to select an entry from your list which may be totally inappropriate for the field for this contact record..*

option that one is faced with. Notice that I use the term option. Because this is a checkbox, you have the option of setting it to **True**, **checked**, or of setting it to **False**, **unchecked**. This option is checked in its default state. In its most simple terms, the default setting means that GoldMine should allow the user to leave the field empty when they are leaving the edit mode of the field. You must remember that GoldMine performs its valid checking as one is leaving a field that has been placed into the edit mode. For you developers, this is done through the **Valid** method in the code usually, however, there are other methods that could have just as easily been utilized. If you should choose the opposite, then a user would be required to enter something into the field that has been placed in the edit mode. This option setting, unchecked, by itself, only means that something must be entered into the field, and GoldMine really doesn't care what that something is.

However, should you combine the first option, unchecked, with the ☐ **Force valid input** option, then you would have begun to take control over the information that is allowed into the field. By default this is unchecked, and, in this state, GoldMine would allow any entry into the field that is currently being edited. Check this box, and GoldMine will require that whatever is entered into the field must also currently reside on the F2 Lookup list. If this is too stringent, then you may want to check the next option, ☐ **Insert closest match**, which will become available as an option once you have selected to ☐ **Force valid input**. This is an out, so to speak, in that, if one begins typing, and nothing on the list matches the entry, then the closest item, alphabetically, from the list will be inserted into the field. I have discovered that selecting this option could produce some very odd field data entry. I would rather see you supply your own out, and not use the insert closest match option at all as someone may enter the closest match, which is not at all correct, and easily not even realize that they have done so. End users, in my opinion, tend to not look at what is actually being entered or, for that matter, look at and read warning messages that are thrown at them.

☐ **Capitalize first letter** is the next option, and may or may not be appropriate for your usage. You need to understand that this is not capitalizing the first letter in the first word, it is capitalizing the first letter of every word in the field. If you can configure your GoldMine so that the user will be allowed to pick entries from the lookup list most of the time, then this option will only control those few times that the user types data into a field by hand. As long as you understand that the first letter of each word in the field will end up capitalized, and this is the result that you are striving to achieve, then go ahead and check this option.

☐ **Pop-up when selected** is an option that is both good, and annoying at the same time. If you should select this option, which is by default unchecked, then every time the user places the associated field into the edit mode, and, as importantly, the field is blank, GoldMine will pop-up the lookup list. You may wish to do this to force the user to realize that there is a set of standardized entries for this field. The annoyance factor comes into play when the user wants to type into the field. If this option is selected, they would be required to click on the **Cancel** button on the lookup list before they could begin typing. I would think that you would want to listen to a few complaints about this in order to achieve the consistency factor that I discussed earlier. If there are too many complaints, then you may opt for an alternative solution such as the ☐ **Auto Fill** option that I will discuss shortly.

☐ **List existing data** is an option that was Added new to GoldMine Premium 8.5. First off, let me state that selecting this option will disable quite a few of the options on the **F2 Field Setup** screen. Now let me explain what it actually does do. Have you ever typed in a Source field value that was not contained within your F2 Lookup List? Selecting this option executes a SQL Query that will pull all of your unique values from the field in question, and populate your F2 Lookup List with the results of this query. So that everything that you have ever typed in by hand will now appear as a possible choice from your F2 Lookup List. Personally, I'm not sure how much value this adds to the F2 Lookup List.

The next three options I will discuss as a group. These three options do not, in any way, have anything to do with how the field is handled, but, instead, determine if you will ☑ **Allow adding**, ☑ **Allow editing**, or ☑ **Allow deleting** of the lookup list items by a user not processing Master Rights. It is important that you understand that these affect the lookup list and its entries, and do not at all affect the field that is being edited. I can't tell you how many times I have heard someone exclaim, *"We have discovered a bug, we had unchecked allow deleting and the users were still able to delete the information in this field."* In trying to achieve as much consistency as is possible, I always recommend that the GoldMine Administrator manage the lookup list in a corporate and consistent manner. In that vein, I do recommend that all three of these options be unchecked such that the users will not be allowed to add, edit or delete items from your lookup list. I further recommend that you establish a standard operating procedure by which the end user is able to notify the GoldMine Administrator any time that they would like an item added, edited or removed from a lookup list.

Tip

☐ ***Auto Fill** is especially useful if you have excessively large F2 Lookup lists.*

☐ **Auto Fill** is the next option that we will want to take a look at. If you have received too many complaints about the pop-up feature, and are truly tired of listening to them, then this may be the option for you. Once the option is selected, whenever a user begins typing, the closest matching entry from the lookup list is inserted into the field. If the user continues typing, and you have not selected to force a valid input, then the users entry will be accepted in lieu of the suggestion from the lookup list. If, however, the user likes the suggestion from the lookup list, they simply have to stop typing, and hit

enter to select the suggested entry. This option is a more subtle approach to consistency. One point that I would like to make, either select the pop-up option or select the auto fill option, but never both options. For our clientele, and when appropriate, I always opt for the auto fill option over the pop-up option. One way to assist in the usage of the auto fill option, is to have the user make sure that their **Tools** | **Options** | **Record** tab option of ☐ **Select contents of fields** is checked or, better yet, by setting this for your organization via a User Override. By default this option is not selected. Please review Chapter 3 of this book for the individual Option settings. And that, believe it or not, was the last option for the **Fields Settings** frame.

Now, mid screen, there is a stand alone item to ☐ **Import lookup entries from another field**. This feature was often requested by end users in earlier versions of GoldMine where it did not exist. Let's just say that you were creating a lookup list for a user defined field of **Contact2.uBillState**. You already have all of the entries for this lookup list contained in the lookup list for the field of **Contact1.State**. You really would not want to have to type in that entire list all over again for the new field. That's exactly where this option comes into play. By selecting this option, you will activate the drop down combo list selection below this option, from which you may choose the field that contains the lookup list that you wish to have copied into the current lookup list. The lookup list will be populated from the selected field once you click on the **OK** button for this dialog screen. It should go without saying that clicking on the **Cancel** button will negate any changes that you have made in the **F2 Field Setup** dialog screen.

Referential Lookup List

Now that we are familiar with the F2 Lookup List **Basics**, I can move on into **Code-based** F2 Lookups.

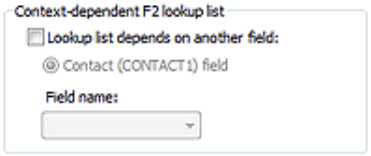


Figure 5-3

Wait, what's that we see in the dialog form? **Contact-dependent F2 lookup list** is new to GoldMine Premium 8.5, and refers to the GoldMine referential lookup list that we have all been asking for since 1988 when GoldMine was first released. Can you believe that we finally have it? I'm thinking that we'd better talk about Contact-dependent F2 lookup list before we move onto the Code-based F2 Lookups. Criminy, we've been waiting for this long enough. Notice, Figure 5-3, that I have pulled out this frame independently for a closer look. One of the first things that I must say is that this frame will look different depending upon the area of GoldMine from where the **F2 Field Setup** screen is rendered. Currently we are on the **Source** field, and this is the frame that is presented to us from the F2 Field Setup screen for that field.

In order for us to use this frame, we must first enable the frame, and we do that by selecting the ☐ **Lookup list depends on another field:** option. The next option, in this case, is not really an option at all in this particular rendering of the F2 Field Setup screen. Use it as a radio button selection, however, there is only one value that can be selected for the field, hence, there is no need to enable the feature as there is nothing that the user can select other than that which is already selected. For the Source field, your only possible table for dependency is the ◉ **Contact (CONTACT1) field**, and the **Field name:** list should now be enabled as well. For this discussion, I'm going to select the **Key1** field in the drop list for Field name:. After having done this, you correctly observe that absolutely nothing has happened. Now isn't that strange? Well, actually no, as this is a multistep process.

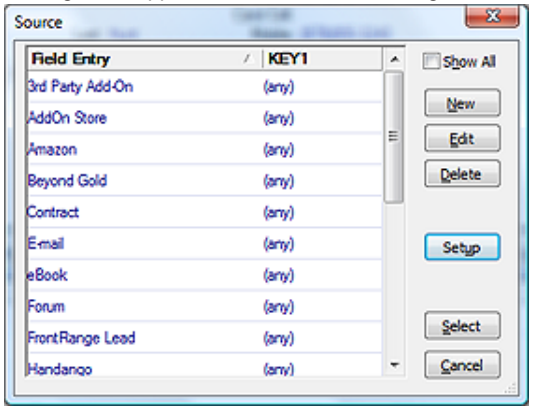


Figure 5-4

Let's click on the **OK** button, and look at the resulting F2 Lookup List dialog form for the Source field now as shown in here Figure 5-4 and compare it to the F2 Lookup List dialog form shown to you previously in Figure 5-1. There is quite a bit of difference now, isn't there?

Instead of having a single list column, you now notice that there is two columns available with the second column being the column that we selected previously on the Setup dialog form which, in this case, is the Key1 field. Currently this column shows the same value, **(any)**, for all list items. So is this any different really than what we had before? No it is not, at least not without further editing. Go ahead, Highlight the first list item and click on the **Edit** button to produce the dialog form shown here in Figure 5-5. From the **F2 Entry** dialog form, one now has the ability to relate this list item to a value contained in the Key1 field. All that one needs to do would be to select the ☐ **Only valid when [KEY1] contains:** option, and then select one of the fields from the Key1 F2 Lookup List or by sim-

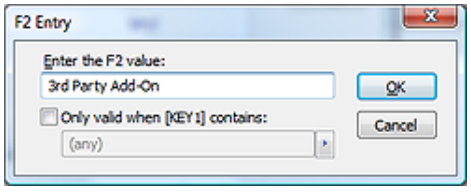


Figure 5-5

WARNING

*While in the development mode, you want to make sure that you keep the ☐ **Show All** option selected as this will keep all of the list items displayed regardless of their relationship. Once you have completed development, you will probably want to reverse that option by unselecting the ☐ **Show All** option.*

ply typing in a value. Once you have done this for all of the items in the current F2 Lookup List, they will then be referentially related and displayed based on the value contained within the chosen field. You asked for it, and you've got it, albeit 20 years later.

As long as the now visible ☐ **Show All** option is not selected, your F2 Lookup List will now display only list items that are related to the Contact1 field name that you have selected. If you ever have a need to display all list items, as when in development mode, make sure that you select the ☐ **Show All** option to regain the entire list.

Let's send out a round of applause to the FrontRange Developers for finally including **Contact-dependent F2 lookup list** in the GoldMine product which I have chosen to rename to **Referential Lookup List**.

Code-based

All right then, now we can catch up with the other books, and move on to the **Code-based** F2 Lookup List. These are the lookups that are associated with the scheduling of and/or the completing of an activity. GoldMine labels the activity code field (**Cal.ActvCode**), simply **Code:**, while it labels the result code field (**ContHist.ResultCode**), **Result:**. The Result field is only available when you are completing an activity, while the Code field is available when you are either scheduling or completing an activity.

Tip

*I cannot emphasize enough the importance of the **Code** and the **Result** fields. Their usage is most noticeable when attempting to create reports. Each report has an **Options** button associated with it. In the options dialog area, you can filter your data on a number of things such as activity type, date range, and, what I am interested in at this point, the **ActvCode** and the **ResultCode** codes. Plan your codes well, and assure that your users are implementing them to assist in your reporting granularity.*

Let's start by looking at the **Code:** field, and it's entries. In fact, let's look specifically at the **Next Action Activity Codes**. Here's an example right from my GoldMine, Figure 5-6. I show you this example only so that you may understand that what I say, and what I do, are not necessarily the same. In other words, I don't always listen to that which I preach. You'll see this as I continue on.

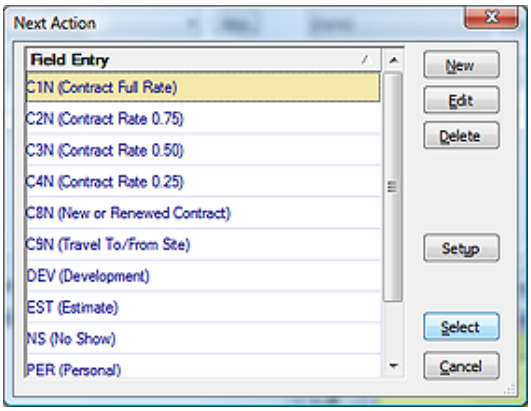


Figure 5-6

One must understand that **Code-based** fields are limited to **only** accepting a maximum of 3 characters. I want to emphasize this fact. Everything you see in Figure 5-6, beyond the first 3 characters, is only there for the purpose of code usage clarity to the end users. Therefore, any note indicators (*//*) would be clearly extraneous, and do not need to be used in these types of code-based lookup lists, however, not using them in GoldMine Premium 8.5 will throw a Warning message as shown here in Figure 5-7. Clicking on the **OK** button will still add the entry to the list however. I will discuss those note indicators in detail later on when I discuss **Text-based** F2 Lookups.

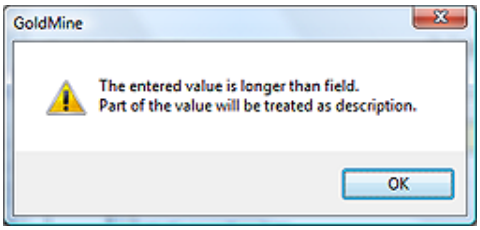


Figure 5-7

I cannot stress enough, if **Code-based** F2 Lookups are going to be used, as they most often are to aide in reporting, then they must be unique per activity. Emphasis is on the *"unique per activity"* part of that statement. You may use the code over again in a different type of activity as these are differentiated further by **RecType**, however, never use a code more than once in any single activity.

Note

*If you are a support based organization, you may want to use the very same **Contact Updater** application that we use in house.*

So, you might ask, *"What are appropriate codes?"*, and that is a very suitable question to ask. Figure 5-6 above, shows my employees choices for Next Actions as we use our Next Actions to track our clients **GoldMine Support Contract** time. Specifically, the first 6 items on this list are unique to my add-on application which calculates Contract Support time plus or minus based on an ActvCode beginning with C, and having a ResultCode of COM. If you are going to be reporting on how many support activities were made for a specific customer these may be appropriate. May I suggest that you would be interested in reporting on the various marketing campaigns that you have on going. Therefore, an appropriate use would be codes for the various campaigns. That way, you could easily report on how many activities were made for any given campaign, or how many sales were directly a result of a given campaign. This seems to me to be a more appropriate use of the activity code field. Using the codes shown in Figure 5-6, I could easily run reports that can be used to invoice a client for billable work accomplished. I could easily run a report to show our clients how their contract time was used, and, I might add, this is often requested by our clients. *"We couldn't have used that much time. Where did all our time go?"* I can't tell you how often I have avoided arguments just by printing and sending these targeted reports, and it was through the careful use of coding that made these reports easy and even possible at all.

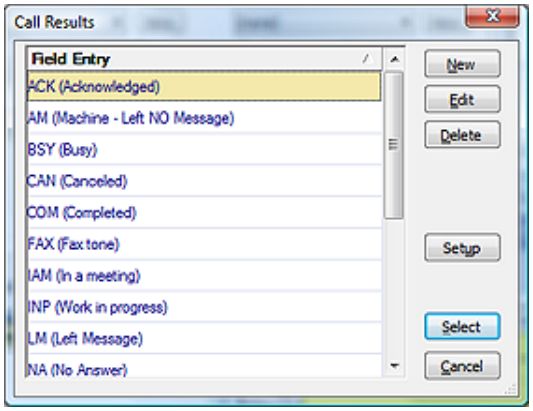


Figure 5-8

I won't look too closely at the **Result** field from the completed activities. They certainly don't hold the same significance as the **Code** field, but they do hold their importance when running reports. For example, I may not want to know all the calls for a given campaign as much as I would want to know all of the calls for a given campaign that were completed (**COM**) successfully. A call that was made where the caller left a message (**LM**) may not be as useful in your report. Using the **Result** field will further help you quantify your reports.

I show you Figure 5-8 for an important reason. This is the F2 lookup from the Complete Unscheduled Incoming Call activity.

Tip

For a multi use F2 Lookup List it may be helpful to annotate the codes to assist with your end user training:

ACK //Acknowledged OC
AM //Machine Left No Message OC
BSY //Busy OC
CAN //Canceled OC
COM //Completed SC
FAX //Fax Tone OC

Where:

IC = Incoming Call Code
OC = Outgoing Call Code
SC = Shared Call Code

You might ask, *"How can an incoming call be busy or have no answer?"*

You might think that these are inappropriate codes to have in this type of activity, and you would be correct except that a few of the F2 lookups in GoldMine are shared among multiple activity types. This particular F2 lookup is shared between Complete Scheduled Call, Complete Unscheduled Outgoing Call, and Complete Unscheduled Incoming Call. This means, to maintain consistency, that you will have to be extra careful in your end user training sessions. You must make it clear when it is appropriate to use a code from a shared list, and when it is not. You may even find that explaining which activities the codes are appropriate for, as a note on the list item, to be more than beneficial when using this type of shared lookup list.

I thought it only appropriate that I should give you a little insight as to how this information is stored. All of the lookups are stored in the shared table **Lookup** in your database. Each field has its own lookup (with exceptions as discussed above) and, hence, its own lookup value. In the **Lookup** table, there are three fields of concern, **FieldName**, **LookupSupp**, and **Entry**. Let's examine the **Call Results Codes** data.

The header record for **Call Result Code** is in:

```
Lookup.FieldName = 'MCALLRSTCDH'
```

Notice that the 11th position contains the letter **H** for **H**eader. This record contains all of the **Setup** settings for this Lookup in the **Lookup.LookupSupp** field. A typical setup might be:

```
Lookup.LookupSupp = 'NNYNNNNNN'
```

While the **Entry** field contains the Lookup window title.

```
Lookup.Entry = 'Call Results'
```

New to GoldMine Premium 8.5.x is the **Lookup.MasterValue** field which is used to support the new **Referential Lookup List**. In its default state it would look as:

```
Lookup.MasterValue = '(any)'
```

So where are the actual values that are displayed in the list? They are stored in subsequent records that can be located by:

```
Lookup.FieldName = 'MCALLRSTCDV'
```

Notice, again, the 11th position is a V for Value. To pull all of this together, one would simply need to execute this SQL Query statement from the query window:

```
select *
from Lookup
where FieldName like 'MCALLRSTCD%'
order by FieldName,
  Entry
```

You programmers will be especially interested in this. Whenever I want to read tables to populate my own forms, I **never** use the API query. I always use a select query via an ODBC connection. I have populated the lookup list of many a form directly from the **Lookup** tables in GoldMine.

WARNING

*If you are utilizing the **Merge:** field, you **must** be certain that only unique characters are place into each byte. If a character/digit/symbol has been utilized once in the field, then be certain that it can't be entered a second time as I have done with my expressions.*

~trim(strtran(MergeCodes,[E],[]))+[E]

*This expression first removes the **E** in any byte in the **Contact1.MergeCode** field before replacing it with another **E**.*

Before I continue on and discuss Text-based lookup lists, I want to discuss a lookup list for a field that would normally be a text-based list, but that I feel should be utilized as a code-based field. That would be the lookup list for the **Merge:** field. You see this is a very important field in GoldMine, yet it only contains 20 character bytes. Hence, I'm going to suggest that you utilize each of those 20 characters as a code for a particular type of merge. Take a look here at Figure 5-9. This is my **Contact1.MergeCode** field lookup list. You'll notice that I have used special characters that I will discuss with you shortly. The significance here is, if I select all 3 of the expressions on the list, that the resulting value in the **Merge:** field will be EMN which only utilizes 3 of the 20 available character bytes in my **Contact1.MergeCode** field. This then means that I have 17 other possible character codes that I can enter, and that can be utilized in GoldMine when the **Merge:** field is used as a filtering condition. You may not see the significance today, however, as you add more and more campaigns, you may find this approach extremely useful.

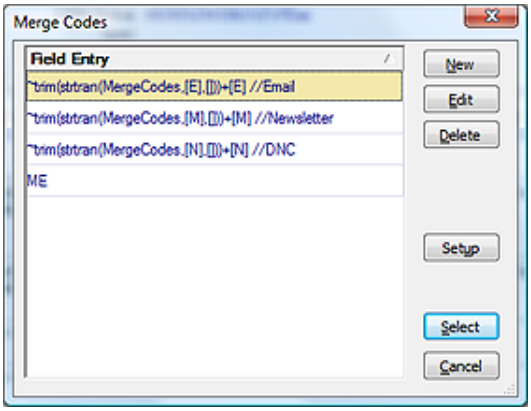


Figure 5-9

Text-based

I discuss the **Text-based** F2 Lookups separately from the code-based lookups although the text-based lookups could certainly be used as code-based, if properly set up. The difference is that I would need to define the coding paradigm for text-based lookups as I just did above for the **Merge:** field. Each field in GoldMine has a **Text-based F2 Lookup**. If the field is a user defined field, the field itself can be whatever length you want it to be, but the lookup will still be bound by the constraints of the lookup table structure. I have discussed these constraints earlier in this chapter for you.

Let's look at some of the special characters that can be used in a text-based lookup. There are now only three symbols that hold special significance to GoldMine when incorporated in a text-based lookup. They are the semicolon (**;**), paired forward slashes (**//**), and the tilde (**~**). Old time GoldMine users will remember the percent (**%**) sign functionality. In GoldMine Premium that functionality has been removed at the behest of the end users who had a need to select a percentage (i.e. **35%**) from the F2 Lookup list, and have the selected item entered into the field.

Note

*The semi-colon (**;**) is only useful for field presentation. If you are using the field as a code-based field, you will probably not want to waste 2 bytes of the field on a space and a comma.*

Tip

*Many administrators have wished to have their F2 Lookup List display in numerical order as opposed to alphabetical order. Whether you are working with **Code-based** lookups, or **Text-based** lookups, this is not currently possible. The list will always display sorted in alphabetical order, and there is no facility to change this order.*

To have numbers sort properly, you must use all possible characters. i.e. 001, 010, 100, etc.

Note

New in GoldMine Premium 8.5 is that the paired slashes (//) no longer display in the F2 Lookup List.

Pre 8.5 Display Example:

Waldo Emerson //Owner

Post 8.5 Display Example:

Waldo Emerson (Owner)

; When you are creating your own code-based system using a text-based lookup, you might, at first, be happy for the functionality supplied by the semicolon (**;**). GoldMine reads the text that you select from the text-based lookup, and if it encounters a semicolon at the end of the string, it interprets this as an append instruction. If there is nothing in the field to begin with, the append instruction is ignored. If something exists in the field, and the user selects another item from the text-based lookup with an append instruction, the second selection is appended to the first, separated by a comma and a space.

Let's start with the field as shown in Figure 5-10, and the associated text-based lookup as shown in Figure 5-11. If you were to select all of the items shown in Figure 5-10, the resulting value in the field would appear as shown in Figure 5-11. You could continue on indefinitely, or, at the very least, until you had reached the end length of the field as had been defined by the user or default by GoldMine.

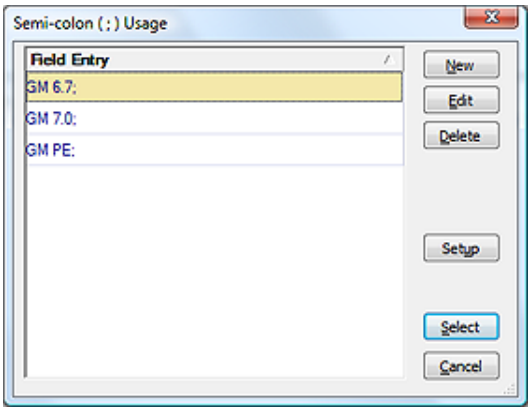


Figure 5-10

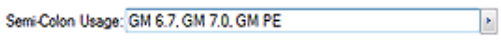


Figure 5-11

// As I have already employed them, Figure 5-9 on the previous page, I suppose that I should discuss the paired forward slashes (*//*) next. You may have noticed that, upon selection of an item from Figure 5-9, that only the information up to (xxx) was entered into my field. We already know that the special characters do not get added to the field, but instead, they are translated to an appropriate function. *What then happened to the rest of the information?* The paired forward slashes in the string, indicates to GoldMine that anything that follows is to be considered as informational only, a Note. GoldMine will not enter either the special character, the paired forward slashes, nor any trailing text, contained in the same string, into a field value. There is an exception to this rule that I will discuss shortly, however, this

is quite a nice feature as it stands. Now you are able to create our own code-based information in a text field using a Text-based F2 Lookup list, and you can annotate the meaning of each code to assist your end users, refer to Figure 5-9. Again, you must be mindful of the 28 character display limitation of the lookup list itself. Additionally, you should never use the paired forward slashes in a Code-based F2 List. They are not necessary anyway as those fields will only hold the 3 character maximum field length of the code.

% The percentage sign possessed a similar functionality to that of the semicolon (**;**). However, alas, its functionality has been removed from your GoldMine Premium product. Too many individuals complained of their inability to use the % in the F2 Lookup list as a **%** for the value field like **35%**. So one has to ask them self: If the functionality was good, and it was, then why did the developers simply not utilize another special character? Possibly the carrot (**^**) symbol. I wonder if we will ever get an answer to that question?

Note

With GoldMine Premium, you no longer have to include the table alias when using the Contact1/Contact2 fields in your expression. This affords you more character bytes for your expression.

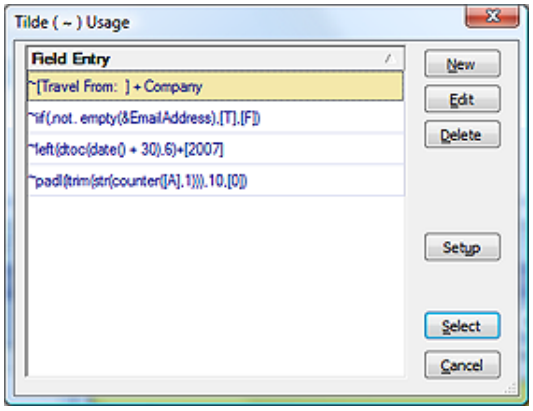


Figure 5-12

~ The tilde sign instructs GoldMine to evaluate what follows as it would a function, and to place the resulting value into the associated field. I have supplied four examples of functions that may be incorporated into a lookup in Figure 5-12, as well, there are examples shown in Figure 5-9. I would emphasize, however, that these expressions could just as easily be used via your Lookup.ini, your reports, your templates, and other locations within the GoldMine environment. Your only limitation, when using a function via the F2 Lookup list, is the 40 character limitation of the field itself.

In this, my first example, as depicted in Figure 5-12, you can see:

~[Travel From:] + Company

WARNING

I have found, in the GoldMine Premium version, that this macro does not always pull the expected information:

~[Travel From:] + Contact

I have found, if scheduling from the record using the menu that the &Contact macro works appropriately. However, if you are scheduling from the Calendar, where the Contact record is not then active, that macro pulls the first alphabetically positioned contact name. In my case it pulls Andrea Dominquez many times when I schedule using this macro from the Calendar.

This might be used in the **F2 Lookup** list for the **Reference** field of an **Appointment** activity for instance. This is a good example of combining character string population with the information contained in a GoldMine field. As I used to charge half rate for travel to and from clients, I always documented that time using **Appointment** activities, and, yes I always utilized the Code field. **C9S** is, in fact, the Code that I utilized. But I digress. This expression was often used to save typing, and worked well when reporting on said activities as well as when Publishing GoldMine calendars to the web that were meaningful. The result, as it might look, of selecting this expression is shown here in Figure 5-13.

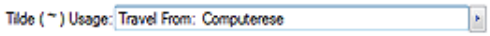


Figure 5-13

My second example will stuff a character based field with a **T** for **T**rue if there is a Primary E-mail Address for the contact record, and with an **F** for **F**alse if there is no Primary E-mail Address for the contact record. This function would best be employed in the Lookup.ini upon NewRecord creation. The expression is:

~iif(.not. empty(&EmailAddress),[T],[F])

The results of the evaluation are displayed in Figure 5-14. This is a good example of the **iif(logical expression, true, false)** function (refer to Appendix A) which can be most useful, as I've shown you, in the GoldMine field attributes.

Figure 5-14

My third example stuffs a character based field with the character representation of the computer system clocks date plus 30 days. The results of this expression:

~left(dtoc(date() + 30),6)+[2009]

generated today are shown to you here in Figure 5-15.

Figure 5-15

My last equation generates a 10 digit sequential number (refer to Appendix A). This can not be used as a unique number, but is the first step to creating a unique number. See the Lookup.ini chapter of this book for a more detailed explanation of use of this function to generate a unique number when in a synchronization environment. The function is:

```
~padl(ltrim(str(counter([A],1))),10,[0])
```

The results are displayed in Figure 5-16. However, I want you to notice that I have no spaces in the expression. Why is that do you think? Why is it that I used a single character **A** as the variable name? The limitations of the F2 Lookup list field for this value is 40 characters, and, if you'll count the characters utilized in my example expression, you will notice that I have employed each and every available space for this expression. There is absolutely no room available for extraneous characters like spaces, and longer variable names.

Figure 5-16

I have completed my description of the various implementations for your F2 Lookups, but I would like to take the time to reemphasize a point. The effective use of the F2 Lookups is your best vehicle to consistency of data input into your GoldMine tables. Make sure that you take the time to develop as many of the lookups as thoroughly as possible. I can not stress enough how much this will help you with any database activities such as filtering, or reporting. Time well spent now, will result in time not being wasted later.

Further, the mere developing of adequate F2 Lookups does not ensure consistent input, only the capability to have consistent data input. You must assure that your end users are well schooled in the use of the F2 Lookups, and it must be clearly understood that the use of the F2 Lookup option should be employed before the option of typing in the data.

Products

Even though FrontRange has removed the **Invoice** capability of a **Forecasted Sale** as of the GoldMine 7 product release, they have maintained the **Product Pricing** that Elan Susser incorporated into the code many years ago for a special client of his. As it is still functioning in GoldMine Premium, I thought that I should like to include it along with the F2 Lookup List capabilities presented here in this chapter.

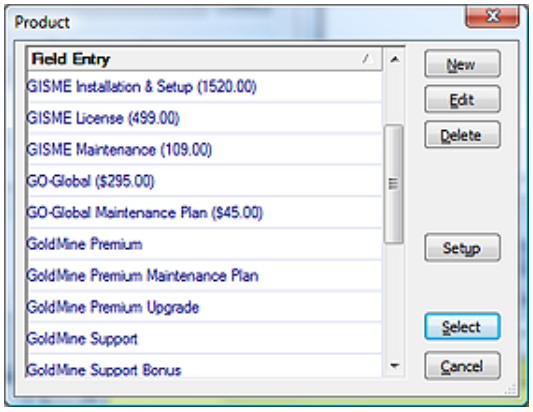


Figure 5-17

I now need to discuss that special usage of the paired forward slashes as was previously discussed. Many GoldMine versions ago, a special feature was added to one particular F2 Lookup List. This functionality may disappear from future builds of GoldMine, however, for now at least, that functionality still exists in the **Product:** field when you **Schedule** a Forecasted **Sale**. New with the GoldMine Premium release, this feature is now extended into the F2 Lookup List associated with the Product field when one is **Completing** a **Sale...** as well.

The functionality was intended to allow a GoldMine Administrator to supply a list of products along with their associated per unit cost. Let's say that I selected, and highlighted the first list entry in Figure 5-17. Not only would the **Product:** field contain the text, **GISME Installation & Setup**, but the **Price:** and **Amount:** field of this forecast sale would also have been populated with **1,525.00** and **1,525.00** respectively. This proved to be highly useful. Instead of supplying an associated price list that the end user could easily lose, and would have required updating for all users, the GoldMine Administrator need only update the pricing in two places (remember that the Forecast Sales and Completed Sales each have separate a F2 Lookup List). The prices would immediately be available to those on the network, and it would also be synchronized to those remote users when next they synchronized. As useful as this feature is, it was, and is, not a highly publicized feature of GoldMine. In fact, I don't know that it is publicized anywhere except in The Hacker's Guide series of books and now this, The Definite Guide, series of books.

Unfortunately, GoldMine Premium does break the compounding capability of the quoting functionality. In previous builds you could have used the semicolon (**;**) to compound the **Product:**, **Price:** and **Amount:** values. You'll remember that semicolon (**;**) was the append function special character. Alas, this capability is gone, and in todays GoldMine, the compounding will only work against the **Product:** field properly, and will not work properly against the **Price:** or **Amount:** fields.

MultiCast Gold

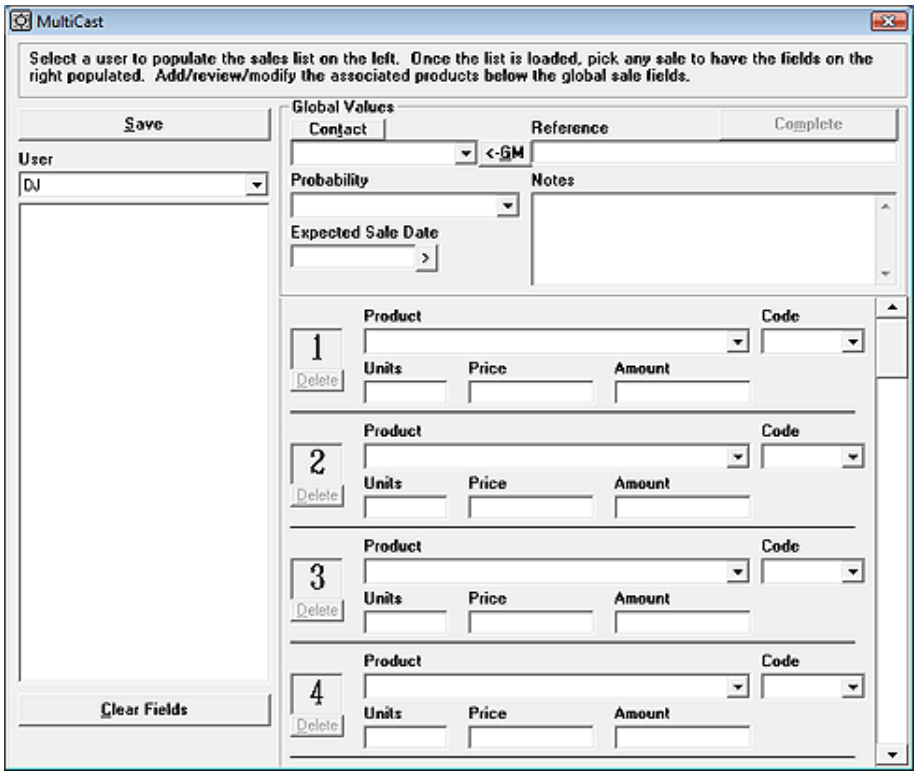


Figure 5-18

Many users had requested the capability of Product line items, and this functionality, which used to exist within GoldMine, is now no longer available. To bring back that functionality while still utilizing the capability of the Product F2 Lookup list functionality, I have designed a product called **MultiCast**. This is a product which I and others sell to end users, that you can purchase and that will work nicely with the GoldMine product. MultiCast will be your Sales Forecasting tool once instituted so you would want to remote **Schedule** | **Sale** and **Complete** | **Sale...** from the users menus in GoldMine. All forecast sales created via this instrument will be shown for each UserID in the column directly below their UserID in MultiCast They may select any of these by double-clicking to modify or complete the activity. The paradigm for this application is that there is one frame for the Global Values, and then there are up to 10 line item products that may be entered. Each field in MultiCast has a lookup list that is pulled directly from your GoldMine F2 Lookup list for the related field. The Product field in MultiCast utilizes the same functionality that exist within the GoldMine F2 Lookup list for that product, and will populate the Price & Amount field in MultiCast with any related values. When the end user Saves this Forecast Sale, MultiCast will create one Master Forecast Sale with a specific ActvCode containing the total value of the Forecast (this is for easy Forecast reporting within GoldMine). As well, MultiCast will create up to 10 related Forecast Sales, each having their own user entered Actv-Code that will represent each of the possible 10 line items.

MultiCast can then be utilized as your Forecast Sale Management Center. We have many corporations utilizing our MultiCast application, and I think, if you have the need for it, that you will certainly like the ease of use that MultiCast offers. By using all of your GoldMines' configured information there is nothing to configure in MultiCast. It can't get any simpler than that. For more information on MultiCast visit www.DJHunt.US.

GoldBook

While we're on the subject of Invoices, and for those of you who are using QuickBooks for creating your Invoices etc.,Computerese Incorporated has introduced yet another new add-on for GoldMine, that being GoldBook. Personally, I am very excited about this application, and I use it extensively myself. GoldBook permit me to use my QuickBooks for my day to day business operations, and will push any of that information into GoldMine for any records that have been linked via the GoldBook application. Additionally, GoldBook permits us to synchronize information between GoldMine and QuickBooks eliminating any need for double entry, hence, typos.

The GoldBook product was designed specifically to replace the defunct QBLink that used to be available from FrontRange, and was specifically designed to surpass the capabilities of QBLink. To that end, I am certain that we have surpassed the capabilities of the QBLink as witnessed by all of the reviews of the GoldBook product.

Specifically, my design goals were to:

- **Eliminate the need to have the User Access Control turned on in any of the Vista operating system environments.**
- **Allow for the easy Linking of existing GoldMine/QuickBooks Customers/Vendors**
- **Allow for the easy Creation/Linking of New GoldMine records as QuickBooks Customers/Vendors**
- **Allow Accounts Perceivable Aging to be automatically brought over from QuickBooks into GoldMine fields.**
- **Allow QuickBooks Estimates/Sales Orders to be automatically brought over to GoldMine Pending/History which will display nicely in the GoldMine Preview windows.**
- **Allows QuickBooks Invoices/Sales Receipts/Purchase Orders to be brought over to GoldMine as History activities which will display nicely in the GoldMine Preview windows.**

I think that I have met all of these design specifications in GoldBook, and here is a sample screenshot showing mismatched linked records between GoldMine and QuickBooks. From this screen you could easily update GoldMine or QuickBooks or both with the simple click of the **Synchronize Contact Fields** button.

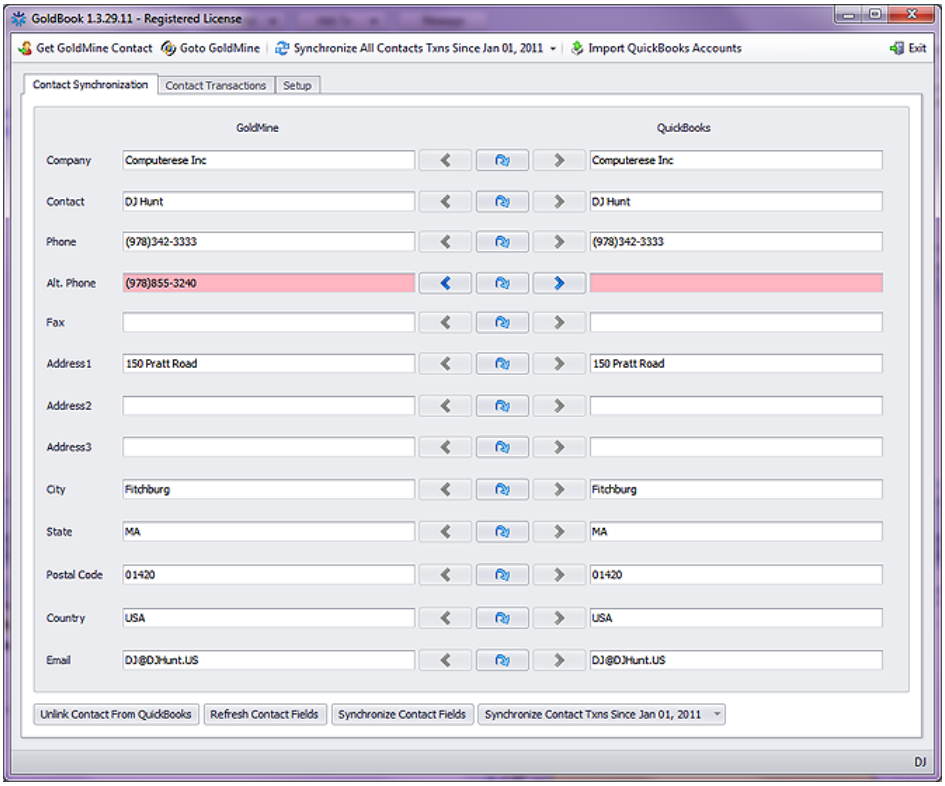


Figure 5-19

Since the original writing of this chapter, I have I have sold the source code for GoldBook to Chris Wettre of W-Systems. The new screen shot above, in Figure 5-19, is actually from the latest version of GoldBook as now is distributed by W-Systems. You can see that Chris has taken GoldBook to an entirely new and better level of integration between GoldMine QuickBooks, while making GoldBook more user friendly. Chris has taken my original GoldBook application to fruition, and for that I am most grateful.

6

In This Chapter

The Basics

Note

In GoldMine Premium, there is no capability of having a single Lookup.ini per Contact set as you had available with the GoldMine Standard Edition.

Rules

A. The Lookup.ini may not exceed 64K in size (theoretically).

B. No single line in the Lookup.ini may exceed 250 characters.

Personally, I have broken the 64K barrier for clients in the XP Pro environment, and the clients claim that the Lookup.ini does function as expected. Your mileage may vary.

This is going to be an easy chapter to rewrite as nothing has changed between versions and builds of GoldMine Premium with regard to this area. The Lookup.ini is how FrontRange Solutions allows the GoldMine Administrators to add external functionality to their GoldMine product. The Lookup.ini is an extremely powerful programming tool that is, unfortunately, not used to its fullest advantage, if at all, in most GoldMine implementations. I can not stress enough, how much control you can gain over your GoldMine with the proper programming of this utility, and, if you have not already guessed, this is my favorite chapter in the entire book. I just love writing this chapter. I hope that you enjoy reading it just as much.

One could write a program that would automatically assign a unique account number to each and every record as it is being created. GoldMine does not have a radio button option, however, you could program (emulate) one into GoldMine via the Lookup.ini. How about the capability to automatically assign new leads to account managers in a rotational manner? This is easily accomplished through the use of the Lookup.ini functionality within GoldMine Premium. Would you like to run an external application, or play a macro when a field changes, when a new contact history record is added, or when a calendar record is edited? You can do all of this as well with the proper programming of your Lookup.ini. How would you like a consistent color coding schema throughout the corporation for calendar activities? Yes, this too is even possible through the Lookup.ini. If you can imagine it, and if it is not already in the GoldMine application, you could probably program it in through the Lookup.ini.

Here is one that I am including in this version of The Hacker's Guide. How would you like to give your users the option of entering **Mr. & Mrs. DJ and Carol Hunt Jr.** into the Contact field, and having the Dear field populated with Dr. & Mrs., the Contact field populated with DJ Hunt Jr., the LastName field populated with Hunt, and finally the Spousal field populated with Carol Hunt?

When thinking of the Lookup.ini, think **Power**.

First of all, it should be understood that there is no Lookup.ini distributed with the GoldMine product. The Lookup.ini is an application that one must create from the get go. I recommend that you create the Lookup.ini using the Windows NotePad application (**Start** | **Run** | **NotePad**). Once created in NotePad, it must be saved as Lookup.ini (see sidebar Tip on the next page) in the root folder of your GoldMine.

The Lookup.ini is broken down into sections. Some of the sections are self contained while other sections are dependant upon the instructions in the **[AutoUpdate]** section of the Lookup.ini. Each section is defined by the header which is defined with enclosed square brackets. Let's start with the **[AutoUpdate]** section which would have a header:

```
[AutoUpdate]
```

I should explain that throughout this chapter I will use white space, and comments freely to assist in the readability of the Lookup.ini application. If you have an exceptionally large Lookup.ini, you will have to consider the rules that must be adhered to (see sidebar Rules) regarding size limitations, and you may have to forego readability formatting for size considerations.

So, what is contained in the **[AutoUpdate]** section? The **[AutoUpdate]** section defines the fields that GoldMine is to watch for changes, and contains a pointer to the section(s) that are to be processed should one of those fields being watched change. There is a single exception to the watch side of the equation, and that is GoldMine allows the **[AutoUpdate]** section to also watch for the creation of a new contact record as opposed to a field, and when a record is created the right side of the equation defines the section(s) that are to be processed accordingly.

Rules

C. The Lookup.ini must reside in the same directory as the GoldMine Premium executable GMW.exe.

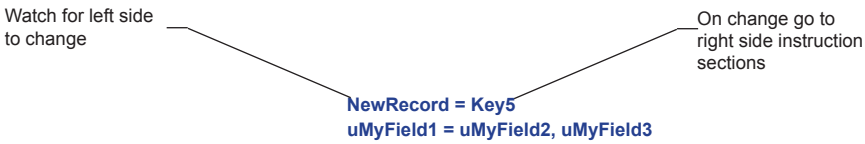
D. You may only have one [AutoUpdate] section in your Lookup.ini.

Tip

*If you are using NotePad to create/edit your Lookup.ini, make certain, when you save the file, that the .txt extension is not put on the file automatically by NotePad. The file name **must** be **Lookup.ini**. If your resulting file is named **Lookup.ini.txt**, you must rename the file back to Lookup.ini.*

Note

The Lookup.ini functions equally well for the Microsoft SQL or the Firebird SQL versions of GoldMine, although I must warn you that I have heard rumors of FrontRange dropping support for Firebird.



If you are planning on formatting a field in a certain way, you can ask GoldMine to watch the field, and then go to the same named section to process the instruction set contained there in. An example of that would be:

```
Key4 = Key4
```

You may wish to add comments to your Lookup.ini. Programmers do this a lot for readability, and for future understanding of what the section was expected to do. If you wish to add a comment to your Lookup.ini, you precede the line with a semicolon (**;**).

Here is an example of what this much of a Lookup.ini might look like:

```
[AutoUpdate]

  ;Watch for a new record being created and process the instruction
  ;set in the Key5 section.

  NewRecord = Key5

  ;Watch for Key4 being changed, and process the instruction sets for
  ;Key4, Key5, and uMyField1 in that order.

  Key4 = Key4, Key5, uMyField1
```

Under the instruction set sections, you must define the action which is to be performed. If the instruction set pertains to a field, then the header must be the name of the field. In our example above, there must be an instruction set for the **Key4**, **Key5**,and **uMyField1** fields or nothing will be processed for the missing instruction sets.

The instruction set itself has sections. In the first section are the lookups. Each lookup is numbered sequentially. In example: **Lookup1**, **Lookup2**, **Lookup3**, ... When processing begins, each lookup will be considered in its sequential order, and all lookups will be considered until one of those lookup variables produces a positive result, or all have been processed with negative results. Consider each of these lookups to be the names of variables. The variable name is on the left side of the equation while the function is on the right hand side of the equation.

Under each lookup is a value list to compare against. If the value in the Lookup(x) variable matches one of the values in the comparison list, the function to the right of the equal sign in the equation is processed. If no match is found, the next lookup variable is evaluated and so on in turn.

Should all of the lookups fail to produce a match, then there can be a fail safe statement. This is called the **Otherwise** statement. If Lookup1 produces a match, process the appropriate expression. If Lookup2 produces a match, then process the appropriate expression. Otherwise, if all Lookup(x)s have returned False, process this expression. You may use the Otherwise statement as a self standing statement without having any Lookup statements.

Note

*Numeric based fields always contain a value. If you are updating a numeric based field through the Lookup.ini, then you **must** include **Overwrite = 1** in the action section.*

The last section is the **Overwrite** statement. This statement is a switch which instructs GoldMine, if there is already information contained in this field, to overwrite the field information with this new information or not depending on the setting of the switch. A zero (**0**) tells GoldMine that it **is not okay** to overwrite the information contained in the field, while a one (**1**) tells GoldMine that it **is permitted** to overwrite the field with the new information.

Updating Fields

In this section, I will put the previously disclosed information into practical use. I will demonstrate a Lookup.ini to update the **Key5** field with a unique account number each time a new record is created. I will populate the **Key1** field with the Account Managers name, and I will populate a user defined field, **uUserName**, with the GoldMine login name for that Account Manager.

Prior to beginning with the Lookup.ini, we need to make some corporate decisions. The **Key5** field must have its **Read Access** attribute set to **(public)** while its **Update Access** attribute is set to **MASTER** or the GoldMine Administrators UserID. I would further stipulate that the user who is creating the new record will be the Account Manager. Lastly, there must be a user defined field created that is called **uUserID**, be a character based field, and be 8 characters long.

When creating the unique account number for each record, one must consider the possibility that synchronization may be occurring. If the field were populated with just a number, then there is the chance for duplication if records are created Server-side and Remote-side as the Counter variable is stored in the Lookup table which synchronizes between the two locations. As I am looking for uniqueness, I must append something to the number, such as the UserID, which itself is unique, that will cause the account number to maintain its uniqueness (most of the time at least).

```
[AutoUpdate]
  NewRecord = Key1, Key5, uUserID

[Key1]
  Lookup1 = &UserName

  DJ = DJ Hunt
  BOB = Bob Jefferson
  BUBBA = Bubba Gump
  Otherwise = Up for grabs

  Overwrite = 1

[Key5]
  Otherwise = &trim(Contact2->uUserID)+padl(ltrim(str(counter([AcctNo],1))),10,[0])

  Overwrite = 1

[uUserID]
  Otherwise = &UserName

  Overwrite = 1
```

Let's take a look at the Lookup.ini section by section. The first section is:

```
[AutoUpdate]
  NewRecord = Key1, uUserID, Key5
```

In this section, the **[AutoUpdate]** section, I am saying that whenever there is a **NewRecord** created that GoldMine should process the instruction sets under the sections **Key1**, **Key5**, and **uUserID**. Just as important, the instruction sets **must** be processed in that specified order. This is done in case the processing in one instruction set is dependant upon the results of the processing in a previous instruction set, better known as cascading instruction sets.

The first instruction set to be processed after the creation of a new record is the **Key1** instruction set.

```
[Key1]
  Lookup1 = &UserName

  DJ = DJ Hunt
  BOB = Bob Jefferson
  BUBBA = Bubba Gump
  Otherwise = Up for grabs

  Overwrite = 1
```

I am using one lookup variable in this instruction set, and I am setting that **Lookup1** variable equal to the resulting value of the GoldMine macro **&UserName**. Here I have used one of the many GoldMine developed macros. Please refer to Appendix B at the back of this book for other available and valuable GoldMine macros. This particular macro will return the logged in users GoldMine **UserID**.

In the next three lines of this instruction set, I am comparing the left side of the equation to the information stored in the **Lookup1** variable. Therefore, if the information in **Lookup1** matches either **DJ**, **BOB** or **BUBBA**, then the value on the right hand side of the appropriate equation should be pushed into the **Key1** field.

Should neither **DJ**, **BOB** or **BUBBA** match the value in the **Lookup1** variable, I have included an **Otherwise** statement. My **Otherwise** statement evaluates if there are no matches found against the value contained in the **Lookup1** variable, and will push **Up for grabs** into the **Key1** field.

Do I require an **Overwrite = 1** statement? Where as, in the past I would have said No, in GoldMine Premium I must answer this question with a definitive Yes. In the past, all but a few of the fields would have been .null. (empty) when a New Record was created within GoldMine. In GoldMine Premium, one now has the ability to input information into any field that exists for the record, hence, for GoldMine Premium and into the GoldMine future, you must utilize the Overwrite statement to account for the possibility that your user may have entered a value into the field.

Emulating Radio Buttons

As you know from a previous chapter, GoldMine only allows the user to add user defined fields in three data types, **Character**, **Numeric** or **Date**. Here's a bit of code that will let you emulate radio buttons within your GoldMine Premium environment using character based fields.

The follow prerequisites are required. You are required to add three new GoldMine fields each being character type, and 1 character in size. They must be named **uRB1**, **uRB2**, and **uRB3** respectively. You should place these on a screen and adjust the **Field Label Size** to **0** while the **Data Size** need only be **2** characters wide. For clarity, you should add an expression field to the right of each field so that it will be easily understood as to which radio button has been selected. You should set up the lookups for these fields to not Allow blank input, and to Pop-up when selected. Additionally, I include one lookup entry, **~chr(169)**, in each F2 Lookup list. A sample of our GoldMine representation is shown here in Figure 6-1.

```
[AutoUpdate]
  uRB1 = uRB2, uRB3
  uRB2 = uRB1, uRB3
  uRB3 = uRB1, uRB2

[uRB1]
  Otherwise = &space(0)

  Overwrite = 1

[uRB2]
  Otherwise = &space(0)

  Overwrite = 1

[uRB3]
  Otherwise = &space(0)

  Overwrite = 1
```

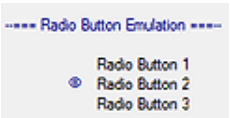


Figure 6-1

Having set those prerequisites, let's look at the individual sections. The last three sections are similar with only variations in field names, therefore, I will only examine the first.

```
[AutoUpdate]
  uRB1 = uRB2, uRB3
  uRB2 = uRB1, uRB3
  uRB3 = uRB1, uRB2
```

The **[AutoUpdate]** section is very straight forward. The first line is instructing GoldMine to watch the **uRB1** for any change, and if it changes, to follow the instruction sets under the **uRB2** and **uRB3** sections in that specific order. The other two lines are just variants of the first so I won't described those in detail.

Now let's examine the instruction set under **uRB2**.

```
[uRB2]
  Otherwise = &space(0)

  Overwrite = 1
```

Tip

Should you ever need to blank out a date based field through the use of the Lookup.ini, I have found that this works nicely.

&.null.

I do not require a **Lookup1** variable field. Not having a **Lookup1** statement causes the processing to drop immediately to the **Otherwise** statement which states: replace the **uRB2** field with nothing (not to be confused with a **.null.** value). I am forcing the value of nothing employing the **space()** function, and I am saying that there should be **0** spaces inserted into this field.

Finally, just in case there was previously a value in the **uRB2** field, I must include the **Overwrite** statement. You'll remember that one (**1**) in this statement lets GoldMine know that it is acceptable to overwrite the old value with the new value which, in this case, is **space(0)**.

Rotationally Assigning Leads to Representatives

Wow! Big section title, huh? Here is the scenario for this Lookup.ini function. This particular office has twelve sales representatives. As new leads are entered into GoldMine, the leads should be distributed to each representative in turn. I have heard this type of distribution of leads to be called a "Round Robin". This is very simple to accomplish using the power of the Lookup.ini, but of course everything is simple once you know how to do it.

The prerequisite for this is to have created a user defined field, **uNextRep**, **N**, **2**, **0**. You do not have to display this field on any screen as we will only be using this field to store the next representatives number. With the field created, we can now discuss the operation of this Lookup.ini.

```
[AUTOUPDATE]
      NewRecord =  uNextRep,  Key1

[UNEXTREP]
      Lookup1 = iif(counter([NextRep], 0) = 12, [A], [Z])

      A = &counter([NextRep], 1, 1, 1)
      Z = &counter([NextRep],1)

      Overwrite = 1
```

```
[KEY1]
  Lookup1 = uNextRep

  1 = DJ Hunt
  2 = Davey Crockett
  3 = Daniel Boone
  4 = Jimney Cricket
  5 = Donald Duck
  6 = Daisy Duck
  7 = Mickey Mouse
  8 = Minnie Mouse
  9 = Goofey
  10 = Pluto
  11 = Pinnochio
  12 = Gippetto

  Overwrite = 1
```

As you can see in the **[AutoUpdate]** section, above, I am only processing these instruction sets upon the creation of a new record at this time. The first instruction set that I process is the **uNextRep** instruction set (sequencing order is critical here). In that instruction set, I set one lookup variable using the **i**mmediate **if** function. This functions syntax is:

```
iif(<Expression>, <True>, <False>)
```

When using the **iif()** function, you must have an expression that can be evaluated to a logical True or False. If the expression evaluates to a True, then the value from the True position is stored in our Lookup1 variable. Alternatively, if the expression evaluates to a False, then the value from the False position is stored in our Lookup1 variable. Our expression is:

```
Lookup1 = iif(counter([NextRep],0) = 12, [A], [Z])
```

Note

*It may be of interest to you to know that all **counter()** variables are stored in the **Lookup** table. The variable name is stored in the first 9 positions of **Lookup.FieldName** with a **C** in the 10th position. The actual counter value is stored in the **Lookup.Entry** field.*

At the time of processing for this instruction set, I am looking to see if the value in the **counter([NextRep],0)** is equal to **12**, the maximum number of representatives for our rotation. If it is equal to **12**, then it is **True**, and I am stuffing our **Lookup1** variable with an **A**. If the value in the **counter([NextRep],0)** is not equal to **12** then I stuff our **Lookup1** variable with a **Z**. As the expression for **Lookup1**, **counter([NextRep],0)** adds **0** to the **NextRep** variable to return the existing value contained in the NextRep variable, hence, I don't have to worry about incrementing the NextRep variable number in the **Lookup** table when I am just testing for what the actual value is presently.

Next, and first in our comparison list, is the comparison statement if the **Lookup1** variable contains an **A**.

```
A = &counter([NextRep], 1, 1, 1)
```

If the **Lookup1** variable contains **A**, then I employ the counter function to reset our counter value, **NextRep**, to its starting position of **1**. My second comparison employs the counter function again, however, this time the counter function is set to increment my counter value, **NextRep**, by **1**. In example, if my counter value were **2**, it would then be incriminated to **3**, and that value would be stuffed into the **uNextRep** field.

```
Z = &counter([NextRep],1)
```

Note

*For a more detailed explanation of the **Counter()** function, refer to Appendix A.*

Notice the difference between the two counter functions. The syntax for the counter function is:

```
counter(<String Variable>, <Increment>, <Start>, <Action>)
```

The Start and Action parameters are optional, and as was shown in the **Z** statement previously. When the Action parameter is set to **1**, then the counter is reset to the number in the Start parameter. When the Action is set to **2**, the counter will be deleted from the Lookup table where the variable, and value of the counter are stored.

Once I have set the **uNextRep** field to a value, I have completed this instruction set. Although I haven't mentioned the last statement in this instruction set, you should remember, from previous examples, its meaning.

According to the **[AutoUpdate]** set, the next instruction set to be processed is the **Key1** instruction set. The **Key1** instruction set is very simple. I stuff the value stored in **uNextRep** into our **Lookup1** variable, and then use that as a comparison value to our list. In example, if the number in **uNextRep** were **4**, then **Jimney Cricket** would be stuffed into the **Key1** field.

As defined, this Lookup.ini will now assign a new contact record to each representative, in turn, until the value in **uNextRep** reaches **12**. It will then reset that number to **1**, and begin the entire distribution cycle again, hence the term, Round Robin.

Last Name Conversion

I am constantly amazed at all of the control that the end user really has at their disposal for the GoldMine product, and how little of it is employed. If you can dream it, you can probably accomplish it through the use of the Lookup.ini.

Let's look at another example now that we're on a roll. This is an example that I wrote many years ago, and FrontRange liked it so much they added to their FAQ's system, albeit inaccurately. The main screen in GoldMine contains a **Contact1.Contact** field, and a **Contact1.LastName** field. For years GoldMine has incorporated the functionality to strip the last name from the **Contact1.Contact** field, and place it into the **Contact1.LastName** field. For all of those years, it has functioned properly in only about 95% of the cases. The functionality, as it exists in GoldMine, is to trim all of the trailing spaces off of the **Contact1.Contact** field. GoldMine then takes everything from the first space in the **Contact1.Contact** field, when reading from right-to-left, to the end of the **Contact1.Contact** field. GoldMine would then put the resulting value into the **Contact1.LastName** field. For a contact name like DJ Hunt, the value of Hunt would be inserted into the **Contact1.LastName** field. However, if that person happened to be named DJ Hunt Jr., then Jr. would be placed into the **Contact1.LastName** field. Not exactly what one would want, is it? This problem occurs on Jr, Sr, II, III, Esq, Rev, and many other variations of the adjective.

As the correct functionality is not within the GoldMine product itself, I must bring the Lookup.ini into play to rectify the situation. Below is my example of a Lookup.ini that will rectify this issue.

```
[AUTOUPDATE]
  NewRecord =  LastName
  Contact = LastName

[LASTNAME]
  Lookup1 = iif([,] $ Contact, [A], [Z])
  Lookup2 = iif((upper(trim(LastName)) == [JR] .or. upper(trim(LastName)) == [JR.]), [B], [Z])
  Lookup3 = iif((upper(trim(LastName)) == [SR] .or. upper(trim(LastName)) == [SR.]), [B], [Z])
  Lookup4 = iif((upper(trim(LastName)) == [II] .or. upper(trim(LastName)) == [III]), [B], [Z])
  Lookup5 = iif((upper(trim(LastName)) == [ESQ] .or. upper(trim(LastName)) == [ESQ.]), [B], [Z])

   A = &alltrim(substr(Contact1->Contact, rat([ ], substr(Contact1->Contact, 1, rat([,], trim(Contact1-
       >Contact))-1))+1, rat([,], trim(Contact1->Contact)) - rat([ ], substr(Contact1->Contact, 1, rat([,],
       trim(Contact1->Contact))))-1))
   B = &alltrim(substr(Contact1->Contact, rat([ ], substr(Contact1->Contact, 1, rat([ ], trim(Contact1-
       >Contact))-1))+1, rat([ ], trim(Contact1->Contact)) - rat([ ], substr(Contact1->Contact, 1, rat([ ],
       trim(Contact1->Contact))-1))))
  Otherwise = &LastName

  Overwrite = 1
```

While this may appear quite complicated, if we examine it in sections, you will see that it is really very straight forward, and easily understood. The first thing that I am accomplishing is to tell GoldMine when to apply the instruction set for the **LastName** field, and that is, quite simply, whenever a **New-Record** is created or whenever the **Contact** field is changed.

Now I get into the meat of this Lookup.ini, the instruction set for the **LastName** field. You will notice that I have used **5** Lookup variables, the last 4 of which are compound iif() functions. Let's examine Lookup1 first as it will be the first one evaluated.

```
Lookup1 = iif([,] $ Contact, [A], [Z])
```

Here I am looking for an occurrence of a comma (,) within the **Contact** field, and if one is so contained, then to stuff an **A** into the **Lookup1** variable. Otherwise place a **Z** into the **Lookup1** variable. If **Lookup1** contains **A**, then GoldMine will compare that to our list and process the instruction set associated with that value. I will discuss this instruction set later in this section. On the other hand, if the **Lookup1** variable contains a **Z**, then GoldMine will not be able to find a comparison match, and upon reaching the end of the instruction set, will loop back to the beginning. At this point GoldMine would begin evaluating the **Lookup2** variable. With minor variations, **Lookup2** through **Lookup5** are identical, and, consequently, I will only need to look closely at the **Lookup2** variable. You are expected to extrapolate the information that you will require to complete **Lookup3** through **Lookup5** from this explanation.

```
Lookup2 = iif((upper(trim(LastName)) == [JR] .or. upper(trim(LastName)) == [JR.]), [B], [Z])
```

Remembering the syntax of the **iif(<Expression>, <True>, <False>)** function, the expression that I am evaluating, and that must return a logical True or False, is a compound expression. Here is the expression:

```
(upper(trim(LastName)) == [JR] .or. upper(trim(LastName))==[JR.])
```

You will notice that there are, in fact, two expressions, and if either of them returns a True, then the entire compound expression is considered to be True because it is an .or. condition. Whereas, if both

Note

*Everything in statement **A** and **B** comparison is a continuous line of instruction and is only wrapped here for presentation purposes.*

Note

*It is important that you understand that the **Lookup1** variable is evaluated first, and only if unsuccessful will **Lookup2** be evaluated, and so on in succession until all of the lookup variables have been evaluated. If all have failed, and if an **Otherwise** clause has been incorporated, then, and only then, will the **Otherwise** clause be evaluated.*

expressions return a False, then the entire compound expression is considered to be False. Now let's examine the first of the two expressions.

```
upper(trim(LastName)) == [JR]
```

Working outward from the inner most parenthesis, I begin by trimming the spaces from the GoldMine **LastName** field using the **trim()** function. Knowing that everyone will type in a different variation of the value in the **LastName** field, I then force the remaining characters to upper case employing the **upper()** function to accomplish this. One might ask here, why am I examining the contents of the **LastName** field? To answer that, I need to look at the sequence of events that are fired by GoldMine. Let's say that you change the name in the **Contact** field. GoldMine will first write this information into the **Contact** field of the record. GoldMine will then proceed to evaluate its own formula for stripping out the last name. This, then, writes the GoldMine interpreted value into the **LastName** field of the record. Lastly, your Lookup.ini will now be evaluated, therefore, instead of trying to find the last name in the **Contact** field as GoldMine would, I can simply look to the **LastName** field for this value.

Next is my operator. Many of you will think that this is a typo, however, it is not. I have used the double equal sign (**==**) purposely. In dBase syntax, the double equal sign operator has special significance. The double equal sign means **exactly equal to**. Well, what is the difference between exact equal to, and equal to? That's a very tough explanation. The former compares the left and right side of the operator at the byte level, therefore, everything on both sides must be equal. That includes the length, and any spaces. This may have been overly cautious on my part as I had already trimmed off all of the spaces, yet I would rather err on the side of caution. The latter compares the left and right sides at the character level. **Test** with three trailing spaces would return a **True** when compared to **Test** with five trailing spaces.

Lastly is the right hand side of my expression, and this contains the value that I will be comparing against, **JR** in this case. The other half of the compound expression has the right side of the expression **JR.** comparing against the left side of the expression. I have tried to capture all of the possible variants that an end user might enter junior into the **Contact** field. To give you some possibilities for your understanding of this scenario, one might expect to see junior expressed as **jr**, **jr.**, **Jr**, **Jr.**, **JR**, **JR.**, **jR**, and **jR.** to name but a few variants. All that I am attempting to do in my expression is to level the playing field, so to speak, such that the expression can be evaluated fairly, and with some expectation of finding matches. You must remember that the left side of the equation is evaluating to upper case, hence, the right side of the equation must show comparison values in upper case as well.

Should my expression contain either **JR** or **Jr.** then stuff a **B** into the **Lookup2** variable. Otherwise place a **Z** into the **Lookup2** variable. If **Lookup2**, **Lookup3**, **Lookup4** or **Lookup5** contain **B** then GoldMine will compare that to our list, and process the instruction set associated with the **B** value. I will discuss this instruction set in detail later in this section. On the other hand, if the **Lookup2**, **Lookup3**, or **Lookup4** variable contain a **Z**, then GoldMine will not find a comparison match, and upon reaching the end of the instruction set, will loop back to the beginning until it has processed through our list comparing the final variable, **Lookup5**.

It is about time for us to examine my list of comparison values. Naturally the first is **A**, and that instruction set looks like this:

```
A = &alltrim(substr(Contact1->Contact, rat([ ], substr(Contact1->Contact, 1, rat([,], trim(Contact1->Contact))-1))+1, rat([,], trim(Contact1->Contact)) - rat([ ], substr(Contact1->Contact, 1, rat([,], trim(Contact1->Contact))))-1))
```

Note

Remember that this instruction is on one continuous line, and wrapped here for presentation only. This line is under the 250 character per line limitation.

The left hand side of the equation, again, is our comparison value, and, if the **Lookup1** variable should match this value, then the expression on the right hand side of our equation would be processed. To try and go through each section of this expression could get rather confusing so I will use pseudo code to try to explain what the function is accomplishing. Mostly, I am using a combination of the **substr(<Field> , <Start Position>, <Number of Characters>)** function, and the **rat(<Character to Look For>,<String to Look In>)** function (refer to Appendix A). I am using these to locate the numerical position of the comma in the **Contact** field, and then to strip off from that position to the end of the string. Here is an example of what might be contained in the **Contact** field.

Note

*I'll be the first to admit that, after all these years, I have devised easier functions to handle this, however, as this is the one that FrontRange has posted on their site, I thought that I should post the corrected **Lookup.ini** here.*

```
Donald J. Hunt, Pastor
```

In the example above you can count, and see that the comma occurs at position **15**. Don't forget to count those spaces and that decimal point. Since the comma occurs at position **15**, I would want to strip off beginning from position **15** on up (I don't want that comma and the space at position **15** and **16** respectively) to the end of the character string. In this example, that would be 8 characters. My substring function would then look like this after all of the internally parentheses had been processed:

```
substr(Contact1->Contact, 15, 8)
```

In the **B** comparison I am doing virtually the same thing only it is a bit more complicated because I do not have a unique character to search for in our character string. I must, therefore, find the numerical position of the first occurrence of a space reading from the right hand side of our character string, and moving to the left until that position is located.

```
B = &alltrim(substr(Contact1->Contact, rat([ ], substr(Contact1->Contact, 1, rat([ ], trim(Contact1->Contact))-1))+1, rat([ ], trim(Contact1->Contact)) - rat([ ], substr(Contact1->Contact, 1, rat([ ], trim(Contact1->Contact))-1))))
```

Again I am using mainly a combination of the **substr()** function and the **rat()** functions to achieve this goal. Looking at another example of the **Contact** field contents:

```
Donald J. Hunt Jr.
```

We can see that my final expression would be:

```
substr(Contact1->Contact, 15, 4)
```

That's a far cry from the 234 character expression shown above, but it is in effect what I am accomplishing in the expression. I wanted to mention the **234 character** length, as you'll remember that I explained earlier, that there is a **250 character limit per line**, and I am bearing down on that limitation. However, you can see that quite a bit of expression can be written within that 250 character limitation. You must always keep in mind the limitations imposed upon you by the Lookup.ini.

Having said all of that, I have to continue with our comparison list. What if none of the Lookup variables had matched? What should GoldMine do then? GoldMine would fail all tests, and in doing so, with no other provision, GoldMine would enter a blank into the field. However, I have foreseen this possibility, and told GoldMine that if all of the Lookup variables fail, then to restuff what you believed to be the last name originally, back into the **LastName** field. I do that with my **Otherwise** clause.

```
Otherwise = &LastName
```

I use the GoldMine macro, **&LastName** (refer to Apendix B), to let GoldMine reprocess the **Contact** field, using its own expression, to produce the characters to be pushed into the **LastName** field.

Finally, because the contents of the **Contact** field may change through the course of time, I must assume that a value was previously contained in the **LastName** field, and, in doing so, I must instruct GoldMine to overwrite that field with the newer information. I do that with the statement that you will, by now, recognize as:

```
Overwrite = 1
```

You must remember that even when a new record is created, that as long as there is something in the **Contact** field, there will always be something in the **LastName** field, and that you must account for that fact in your Lookup.ini instruction set.

Running External Applications

Next, I will examine the use of a Lookup.ini for the running of an external application. One typical example of this need is caused by the limitations of the Lookup.ini itself. One of my clients had territories assigned by zip codes. A Lookup.ini to handle the stuffing of several fields based upon all of the zip codes in the United States would far exceed the limitations of the Lookup.ini (not necessarily true with todays operating systems). Sometimes, in these cases, I have tried to combine zip codes into groups to reduce the size of the Lookup.ini. In most cases, however, this has failed. I could easily create an external application that looks through a zip code database for the proper information, and then stuffs that information back into the appropriate GoldMine fields. Although I don't go into the details of the inner workings of the external application in this book, I must know how to execute that application once it has been created should a field change. Additionally, as I'll explain later, you could have an application run when a new record is created, or when a record is edited in any of the main GoldMine tables. Let's look at the case where an application needs to be run when the contents of the **Contact1->Key1** field has changed.

Note

You may include an AutoUpdate action before running your external application.

Tip

You must include the path to the executable if the executable is not in the windows default search path.

```
[AutoUpdate]
  Key1 = Key1

[Key1]
  Run = C:\AnyPath\ZipCode.exe
  RunFlags = 2
```

This is the piece of code that should allow me to do that. I am watching the **Key1** field for a change, and should any change occur, I am instructing GoldMine to process the instruction set for the **Key1** field.

My first statement under the **Key1** instruction set is to tell GoldMine which application to run. I do this with the **Run** statement. GoldMine requires that I set flags to tell it under which conditions the **Run** statement is acceptable to execute. Obviously it is to be run if the **Key1** field changes, but you must consider that as an **.and.** inclusion. That means that not only must the **Key1** field have changed, but the conditions specified by the **RunFlags** statement must be met as well.

The RunFlags statement value is the sum of the values for three possible conditions. These conditions are:

```
Run when the field's lookup value is found                =    1
Run when the field is updated via the [AutoUpdate] section =    2
Run when the field is updated via Automatic Processes     =    4
```

As I do not have any Lookup values to find, and I do not want the application to run during our running of the Automated Processes, I have chosen to use **2** as the value in our **RunFlags** statement. Should you have a Lookup list, and you desire to have this executable run during your running of the Automated Processes, then you would have to use the value of **7** which is the sum of the three possible values.

In this example I will ask the Lookup.ini to watch for a change in the **Key1** field, and with that change, if the value appears in the lookup list of items to run an application. If the value is not included in the lookup list of values, then do not run the application.

```
[AutoUpdate]
  Key1 = Key1

[Key1]

  Lookup1 = upper(trim(Contact1->Key1))

  DD = Dave Dunlap
  DJ = DJ Hunt
  Overwrite = 1

  Run = Calc.exe
  RunFlags = 3
```

With **RunFlags** set to **3**, when a lookup value is found in the list (in this case **DD** or **DJ**), the field is updated accordingly, and then, after the field update, the **Calc** application is run automatically. Placing any other value in the **Key1** field, and absolutely nothing will happen. The field will not be updated, nor will the external application be run.

Now we need to look at the other method for running external applications using the Lookup.ini. As I stated previously, when a new record is created, **[OnNewRun]**, or when a record is edited, **[OnEditRun]** in any of the main tables, you may have a need to run an external application. These two sections of the Lookup.ini are independent of the **[AutoUpdate]** section, and they are neither controlled by it, nor react to it. Their statements are tested for validity at all times, and should a condition be met, then the associated executable will be launched.

```
[OnNewRun]

  Cal-S = SaleCApp.exe
  Cal-C = CallCApp.exe
  Cal = CalApp.exe

  ContHist-S = SaleHApp.exe
  ContHist-CI = InCallHApp.exe
  ContHist = HistApp.exe

  ContSupp-P = DetailApp.exe
  Contact1 = NewContact.exe

  Otherwise = AnyOldApp.exe
  AppendRecNo = 1
  DisableFromAP = 1
```

The sample above is similar to the one that is supplied by GoldMine. The syntax for the **[OnNewRun]** and **[OnEditRun]** is identical with the exception that the former applies only to newly created records of the specified type, while the latter applies only to those records of a specified type that have been edited.

In both cases you must supply the section header. In the sample case I used **[OnNewRun]**, but I could have just as easily used **[OnEditRun]**. In these sections, one defines the left hand side of the equation with the table and/or the table record type that would required to be added or edited for the application on the right hand side of the equation to be executed.

Upon close examination of the sample, you will notice that the tables are identified only by their names. Consequently, any time a new record is added or edited in the **Contact1**, **Contact2**, **ContSupp**, **ContHist**, or the **Cal** tables, the associated executable will be launched. You must use the table names as they have been identified here.

Tip

If the executable does not reside in the same directory as the Lookup.ini, you must disclose the full path to the executable in your statement.

Alternatively, you may launch an executable based on the addition of, or the editing of a specific type of record in certain tables. Let's look at a couple of the lines.

```
Cal-S = SaleCalcApp.exe
Cal-C = CallCalcApp.exe
```

To identify record types, you must use the table name, a dash, and then the information contained in the **RecType** field of the table (refer to The Tables chapter). In the first statement, above, I use the **RecType** of **S** for sales record. Therefore, the addition of, or the editing of a **Forecasted Sale** within GoldMine will trigger the launching of the **SaleCalcApp.exe** executable. Similarly, in the second statement, I use the **RecType** of **C** for a **Call** record. If a call is scheduled in, or edited from the calendar then the **CallCalcApp.exe** executable will be launched.

Note

*In the **ContHist** table, only the first **2** positions of the **RecType** field are monitored.*

Here are the RecTypes that can be used:

Cal		ContHist		ContSupp	
A	Appointment	A	Appointment	C	Additional Contact
C	Call Back	CC	Call Back	L	Linked Document
D	To-do Action	CI	Incoming Call	O	Organizational Chart
M	Message	CM	Returned Message	P	Detail Record
O	Other	CO	Outgoing Call	R	Referral Record
S	Forecast Sale	D	To-do Action		
T	Next Action	L	Letter		
		M	Message		
		O	Other		
		S	Sale		
		T	Next Action		

There are no **RecTypes** contained in either the **Contact1** or the **Contact2** table.

As with all of the other instruction sets, GoldMine permits the usage of the **Otherwise** clause. In my example I used:

```
Otherwise = AnyOldApp.exe
```

Note

*In both statements, a **1** sets the switch to **True**, while a **0**, or no statement at all, sets the switch to **False**.*

While just below that statement, there were two special instructions for this section. They were:

```
AppendRecNo = 1
DisableFromAP = 1
```

The first, **AppendRecNo**, instructs GoldMine to append the number of the record at the current pointer position in the table as a parameter to the launching executable. In the **Cal-C** statement, if the record pointer were at **RecID BRQ1DSJ#(QQ%R#Y** in the calendar table, then the Lookup.ini launching instruction would look similar to:

Note

*You cannot believe the GoldMine Premium Help files on this, as they are way outdated. Against SQL tables, where there are no record numbers, the **RecID** is passed in lieu of the **RecNo**.*

```
CallCApp.exe BRQ1DSJ#(QQ%R#Y
```

By using a parameter such as I have described, your external application may make use of the GoldMine Dynamic Data Exchange (**DDE**) capability to locate the appropriate record in the **Cal** table, and take action against it and/or extract the information from it. In fact, passing this parameter is the only way for the external application to know where the record pointer is located. In a previous statement, I mentioned that I wrote an application that would populate fields based on the zip code. That application had to be instructed as to which record in the **Contact1** table to add this information into. This is the very method that I employed for the launching of that application.

The final statement employed the switch **DisableFromAP**. This instruction tells GoldMine to disable all of the options in this section if the newly added record, or edited record resulted as consequence of the execution of a GoldMine Automated Process.

Playing Macros

GoldMine allows the users to create keyboard macros, and these macros could be played through the appropriate use of your Lookup.ini. The keyboard macros could be played either through an instruction set or through the **[OnNewRun]** or the **[OnEditRun]** sections of the Lookup.ini. This adds yet another new level of functionality to your Lookup.ini. Let's look at a simple Lookup.ini that makes use of the **PlayMacro(<MacroNumber>)** function.

```
[AutoUpdate]
  Key1 = Key1
```

Tip

Keyboard macros are user specific. If you plan to play a keyboard macro through the use of the Lookup.ini, then you must assure that the macro number that is being played is identical for each and every user in your GoldMine system.

Tip

Keyboard macros, which move from the current locked record to another record, may prevent the calling function from completing properly. Try to avoid situations through the Lookup.ini where the playing keyboard macro changes to another contact record.

Note

Keyboard macros tend to falter after extended periods of use for some unknown reasons. You are advised to periodically refresh the users Keyboard Macros.

```
[Key1]
  Lookup1 = Key1

  West Coast = &PlayMacro(841)
  East Coast = &PlayMacro(840)

[OnNewRun]
  Contact1 = &PlayMacro(835)
```

I first ask GoldMine to watch the **Key1** field for any change, and, if there is a change, to follow the instruction set for **[Key1]**. I then make use of the **Lookup1** variable to look at the current value of the **Key1** field, and to compare that to my list of values. If there is a match against **West Coast**, then GoldMine is instructed to play the keyboard macro number **841** which belongs to the currently logged in GoldMine UserID.

You probably would benefit more if I put this into a real scenario. Suppose that every time a new customer record was added to GoldMine, that you wanted to have that person scheduled for a call back. You would first record your keyboard macro to schedule a call back activity, and save it to one of the supplied GoldMine icons. The first icon in the column of icons is number **800**, while the next is **801**, and so forth on down the column. As you can see above, it is important that you know the number of the keyboard macro to be played. I could now employ the **[OnNewRun]** section of the Lookup.ini file to watch for all newly created records in the **Contact1** table, and as each is created to play the appropriate macro.

If you are not adept at writing external applications, you can usually circumvent that need by creating keyboard macros that will perform most of your actions. You can then have those keyboard macros played through the use of your Lookup.ini. Make sure that you read the sidebar Tips as they contain some important additional information pertaining to this section.

Color Coding Calendar Activities

Your organization may want to standardize their color coding schema for calendar activities. Why it is a part of the Lookup.ini, I do not know, but it is, hence, I will discuss it now. The Lookup.ini can have another independent section called **[CalClrCode]** which identifies your corporate color schema for calendar activities. Here is a typical example of its usage.

```
[CalClrCode]
  A = 3
  C = 1
  T = 4
  M = 0
  O = 2
```

On the left hand side of the equation, you identify the activity that is to receive the specified color that is contained in the coded number in the right hand side of the equation. Below I have listed the coding for each side of the equation.

Left Hand Side	Right Hand Side		
A = Appointments	0 = Bright Blue	5 = Bright Yellow	10 = Green
C = Calls	1 = Bright Purple	6 = Cyan	11 = Yellow
T = Next Actions	2 = Bright Red	7 = White	12 = Blue
M = Messages	3 = Bright Cyan	8 = Gray	13 = Purple
O = Other Actions	4 = Bright Green	9 = Red	14 = Dark Grey

One can clearly see that I have asked for my appointments to be colored bright cyan, while my calls will be colored bright purple on the GoldMine graphical calendar interface. This schema, or the one that you may develop for your organization, will provide another level of consistency in your organization. No matter which users graphical calendar that you are examining, the calendars will be consistently color coded such that everyone will know that it is an appointment if it is bright cyan for instance. A caveat here: Users may opt to change the color when Scheduling or Modifying a Scheduled Activity. Consistency in application usage makes for a better all around use of the GoldMine application itself. Strive for consistency wherever and whenever you can achieve it in your corporate development of the GoldMine product.

But wait, we are not finished. FrontRange has now provided a more granular approach to the coporate color coding schema capabilities.

```
[CalClrCode]
  A = 4
  A-TRG = 5
  C = 1
  C-FUP = 2
  O = 2
```

That's correct, we can now begin to further the **ActvCode** level. For instance: The **A-TRG = 5** statement is stating that if this scheduled **Activity:** is that of an Appointment type, and if the **Code:** is **TRG** then the color coding for this activity should be **5** or **Bight Yellow**. Actually, this capability existed before GoldMine Premium 8.5.1.12, which is the version that I'm currently writing this book against, however, one of my readers brought this to my attention so that I could include it in future books such as now. You readers are sometimes my best resources. Either way this is way cool, because now you can use corporate color coding to segregate those special activities like **Forecast Sales** of **PRO** (Prospects).

Generating Your Own Unique Identifier

Using your Lookup.ini, you could generate your own unique identifier for a newly created record, and have that identifier populate one of your fields.

Why would I want a unique identifier on top of the AccountNo and RecID which are already unique?

GoldMines AccountNo, and RecID employee special higher level characters that are not recognized by all systems. The unique identifier that I am generating employees only standard characters and numbers, and can be more easily used when Importing into GoldMine while trying to match existing GoldMine records for updating.

To accomplish this, I will make use of the **Counter()** function. There is one thing that is important to understand before I begin this exercise. The **Counter()** function stores the number that it produces in the GoldMine **Lookup** table.

Why is this important?

That is also a very good question for you to have asked. If you are synchronizing your GoldMine data, then this number, as it resides in the **Lookup** table at the time of the synchronization, will synchronize out to those remote users. This means, in using this number alone, you are very likely going to generate duplicate numbers if your records are not created from a single location. Someone could be retrieving the next sequential number on the server while a remote user is retrieving that same sequential number on their remote GoldMine. Therefore, you must never use the **Counter()** function by itself to generate a unique number when there is any chance of synchronization within your organization. What I propose, knowing that the user login name is unique within GoldMine, is to append the user login name to a padded form of the counter. This, then, will always generate a unique number for each record even in a synchronization scenario. Here is the code that I propose that you employee to achieve this unique number.

Tip

*As you are generating a unique ID, you may wish to protect the field from changes. You should consider setting the **Update rights:** field, for this field to **MASTER**, or at least to a group of users having **Master Rights**.*

```
[AUTOUPDATE]
  NewRecord =  Key5

[KEY5]
  Otherwise = &&UserName+padl(ltrim(str(counter([AcctNo],1))), 8, [0])
```

That is quite a mouthful, wouldn't you say? Obviously, I am watching for a newly created record, and once GoldMine realizes that a new record has been created, to process the instruction set for the **Key5** field. Simple, now that you know the process, isn't it? I already know that the leading **&** in the **Otherwise** statement lets GoldMine know to evaluate the remainder of the string as an expression. The first part of our equation is **&UserName**, which is a GoldMine macro function (refer to Appendix B). This macro function extracts the trimmed **UserID** login name of the active user as a string. I say trimmed, because, as you are aware, the **UserID** field can contain up to eight characters. The **&UserName** macro only extracts the name from this field, and not any of the additional spaces that may be present.

To the **UserID** login name, I am then appending, working from the inner most parenthesis outward, the **Counter()** function, the **Str()** function, the **LTrim()** function, and finally the **PadL()** function. All of these functions are covered in more detail in Appendix A at the end of this book.

However, in pseudo code, I am:

- retrieving the next sequential number for the variable **AcctNo**, **Counter()**
- converting that number to a string, **Str()**
- trimming off any spaces from the left side of the string, **LTrim()**
- padding the left side of the string to eight characters with the character **0**, **PadL()**

Record Typing (Another Approach)

GoldMine Premium contains the **Record Typing** concept, as I had discussed in Chapter 4. As the GoldMine Administrator, it is important that you take this into consideration when you are developing your Lookup.ini.

In Chapter 4, I had discussed employing the **Contact1->Key1** field for **Record Typing**. I had discussed that this field could identify the underlying record as being either a **Buyer**, **Seller**, **Agent**, or **Property** record. I then discussed using one field to hold different information based upon the record type. Let's say, for this example, that I am going to use the **Contact2->UserDef01** field to hold the

availability date if this is a record type of **Property**. While a record type of **Buyer**, or **Seller**, this same field could contain a **Status**. I am also going to consider in what state the **Agent** may be located, and supply the agents name associated with that state. Let's see how this might look in the Lookup.ini.

```
[AutoUpdate]
  Key1 = UserDef01

[UserDef01]

  Lookup1 = upper(trim(Contact1->Key1))
  Lookup2 = upper(trim(Contact1->Key1))+upper(trim(Contact1->State))

  PROPERTY = &dtoc(date())
  BUYER = Ready to Purchase
  SELLER = Available for Sale
  AGENTMA = DJ Hunt
  AGENTNH = Davey Crackett
  AGENTME = Waldo Fairchild
```

I am simply reminding you that **Record Typing** throws a whole new monkey wrench into the Lookup.ini development environment. As the GoldMine Administrator, you must take this into consideration when developing your Lookup.ini.

Currency Formatting

Many users have asked to have the ability to format a field in a currency format. Many financial institutions, using GoldMine, have asked me for this capability. Here is a little Lookup.ini code that I have developed to accomplish this. To use this code you need a character based field, so I added the field:

```
uDlrFormat, C, 15
```

Here is the Lookup.ini code to keep the Contact2.uDlrFormat field formatted properly:

```
[AutoUpdate]
  uDlrFormat = uDlrFormat

[uDlrFormat]
  Lookup1 = alltrim(str(len(trim(strtran(strtran(Contact2->uDlrFormat, “,”,””),”$”,””)))))
      +iif(“.” $ Contact2->uDlrFormat, “T”, “F”)

  2F = &”$”+alltrim(Contact2->uDlrFormat)+”.00”
  3F = &”$”+alltrim(Contact2->uDlrFormat)+”.00”
  4F = &”$”+left(trim(strtran(strtran(Contact2->uDlrFormat, “,”,””),”$”,””)),1)+”,”
      +substr(trim(strtran(strtran(Contact2->uDlrFormat, “,”,””),”$”,””)), 2, 3)+”.00”
 5F = &”$”+left(trim(strtran(strtran(Contact2->uDlrFormat, “,”,””),”$”,””)), 2)+”,”
      +substr(trim(strtran(strtran(Contact2->uDlrFormat, “,”,””),”$”,””)), 3, 3)+”.00”
  6F = &”$”+left(trim(strtran(strtran(Contact2->uDlrFormat, “,”,””),”$”,””)), 3)+”,”
      +substr(trim(strtran(strtran(Contact2->uDlrFormat, “,”,””),”$”,””)), 4, 3)+”.00”
  7F = &”$”+left(trim(strtran(strtran(Contact2->uDlrFormat, “,”,””),”$”,””)), 1)+”,”
      +substr(trim(strtran(strtran(Contact2->uDlrFormat, “,”,””),”$”,””)), 2, 3)+”,”
      +substr(trim(strtran(strtran(Contact2->uDlrFormat, “,”,””),”$”,””)), 5, 3)+”.00”

  5T = &”$”+alltrim(Contact2->uDlrFormat)
  6T = &”$”+alltrim(Contact2->uDlrFormat)
  7T =&”$”+left(trim(strtran(strtran(Contact2->uDlrFormat, “,”,””),”$”,””)), 1)+”,”
      +substr(trim(strtran(strtran(Contact2->uDlrFormat, “,”,””),”$”,””)), 2, 6)
  8T =&”$”+left(trim(strtran(strtran(Contact2->uDlrFormat, “,”,””),”$”,””)), 2)+”,”
      +substr(trim(strtran(strtran(Contact2->uDlrFormat, “,”,””),”$”,””)), 3, 6)
  9T =&”$”+left(trim(strtran(strtran(Contact2->uDlrFormat, “,”,””),”$”,””)), 3)+”,”
      +substr(trim(strtran(strtran(Contact2->uDlrFormat, “,”,””),”$”,””)), 4, 6)
 10T =&”$”+left(trim(strtran(strtran(Contact2->uDlrFormat, “,”,””),”$”,””)), 1)+”,”
      +substr(trim(strtran(strtran(Contact2->uDlrFormat, “,”,””),”$”,””)), 2, 3)+”,”
      +substr(trim(strtran(strtran(Contact2->uDlrFormat, “,”,””),”$”,””)), 5, 6)

Otherwise = &Contact2->uDlrFormat
Overwrite = 1
```

Note

Remember that each line is a continuous line of code. The lines are wrapped, and indented here for presentation and readability only.

The premiss behind this code is simple. The Lookup.ini is watching the **Contact2.uDlrFormat** field for changes, and, when the field changes, is processing the instruction set for that field (itself). All the coding is based on the length of the characters in the field after the dollar sign (**$**) and commas (**,**) have been removed. In the Lookup.ini, I append that length to a **T**rue or **F**alse letter if the value contains a decimal.

Once I have populated the Lookup1 variable with a value, I assign instruction sets based on that value. For instance, a Lookup1 value of **7T** means that the value in the field, without a dollar sign or commas, is **7** characters long, and the **T** means that one of those seven characters is a decimal. Formatting this number becomes easy. I add back in the $ sign, and add to that the 1st character of the field, and a comma. To that, I then simply add the rest of the characters.

Lookup.ini Razzle - Dazzle

A number entered as **1234.56**, would be processed, and returned back to the field as **$1,234.56** as would this **$1234.56** , or this **1,234.56** if it were entered into that same field. Perfect. This is exactly what your client was looking to achieve.

I have already commented on how important consistency of data in GoldMine is, and how difficult that is to achieve. This section of this chapter will cover a Lookup.ini that I had created for my Financial Institution type of GoldMine usage, however, it contains many examples of the different ways in which a Lookup.ini can be utilized.

Here are the prerequisite fields that are needed, in addition to the GoldMine default fields, so that this Lookup.ini will function properly:

```
UserDef01, C 40
uALabelA, C, 40
uALabelB, C, 40
UserDef02, C, 40
UserDef03, C, 40
uNNCl, C, 20
uNNMa, C, 20
uADearA, C, 20
uADearB, C, 20
uAAddress1, C, 40
uAAddress2, C, 40
uAAddress3, C, 40
uACity, C, 30
uAState, C, 20
uAZip, C, 10
uCTaxID, C, 12
uMTaxID, C, 12
uSSNCl, C, 12
uSSNMa, C, 12
```

Throughout the Lookup.ini, I will use Comment lines to annotate the functions of the various instruction sets. These Comment lines will be in bold black, and they will be preceded by a semicolon (;).

```
[AutoUpdate]
  NewRecord = UserDef01, Company, LastName, Department, uALabelA, uALabelB, Contact, Dear,
              Secr, UserDef02, UserDef03, uNNCL, uNNMA, uADearA, uADearB
  Contact = LastName, Department, uALabelA, uALabelB, Contact, Dear, Secr, UserDef02, UserDef03,
            uNNCL, uNNMA, uADearA, uADearB
  State = Country, uAState, uACountry
  Company = UserDef01, Company
  Address1 = uAAddress1
  Address2 = uAAddress2
  Address3 = uAAddress3
  City = uACity
  uAState = uACountry
  Zip = uAZip
  uCTaxID = uCTaxID
  uMTaxID = uMTaxID
  uSSNCl = uSSNCl
  uSSNMa = uSSNMa

[COMPANY]

; This instruction set will consistently enter a Company name into the Company field.
; If the Company name is entered as The Financial Institution, this instruction set will convert
; that name to Financial Institution, The
; Any Company name that does not begin with the word The will remain as entered

  Lookup1=iif(upper(left(&Company,4))==[THE ], 'A', 'Z')

  A = &substr(&Company, 5, 40)+[, The]
  Otherwise = &&Company

  Overwrite = 1

[CONTACT]

; Here I am looking for any salutation that may have been entered into the Contact field
; and I am stripping it out.  Notice the order of processing in the AutoUpdate section
; above as it is critical to the proper processing of this Lookup.ini

  Lookup1 = iif(upper(left(&Contact, 10)) == [MR. & MRS.], [A], [Z])
  Lookup2 = iif(upper(left(&Contact, 8)) == [MR & MRS], [C], [Z])
  Lookup3 = iif(upper(left(&Contact, 4)) == [MRS.], [D], [Z])
```

Note

Remember that each line is a continuous line of code. The lines are wrapped, and indented here for presentation and readability only.

```
  Lookup4 = iif((upper(left(&Contact, 3)) == [MR.] .or. upper(left(&Contact, 3)) == [MRS] .or.
            upper(left(&Contact, 3)) == [MS.]), [E], [Z])
  Lookup5 = iif((upper(left(&Contact, 2)) == [MR] .or. upper(left(&Contact, 2)) == [MS]), [F], [Z])
  Lookup6 = iif(upper(left(&Contact, 10)) == [DR. & MRS.], [A], [Z])
  Lookup7 = iif(upper(left(&Contact, 8)) == [DR & MRS], [C], [Z])
  Lookup8 = iif(upper(left(&Contact, 3)) == [DR.], [E], [Z])
  Lookup9 = iif(upper(left(&Contact, 2)) == [DR], [F], [Z])

  A = &substr(&Contact, 12, 40)
  B = &substr(&Contact, 14, 40)
  C = &substr(&Contact, 10, 40)
  D = &substr(&Contact, 6, 40)
  E = &substr(&Contact, 5, 40)
  F = &substr(&Contact, 4, 40)
  Otherwise = &&Contact

  Overwrite = 1

[COUNTRY]

; Here I am automatically determining the Country field base on the information entered into
; the State field.

  Lookup1 = iif(trim(&State) $ "MA NY CT VT NH ME RI VA NC TX MI MO KS CA PA FL ND SD HI AL
            KY ID WI DC", "A", "Z")
  Lookup2 = iif(trim(&State) $ "UT NM AZ NV OH IL OR WA OK WV SC DE CO MD GA MN IN NJ IA NE
            TN LA AR MS", "A", "Z")
  Lookup3 = iif(trim(&State) $ "ON AB NS MB QC", "B", "Z")

  A = USA
  B = Canada
  Otherwise = &&Country

  Overwrite = 1

[DEAR]

; Here I am looking for any compound names, and stripping them apart for the Dear field.
; For instance, if the Contact name were entered as DJ & Carol Hunt, this instruction set would
; populate the Dear field with DJ & Carol.

; Pay particular attention to the order of processing in the AutoUpdate section as it is
; critical to the proper processing of this Lookup.ini

  Lookup1 = iif([ AND ] $ upper(&Contact), [A], [Z])
  Lookup2 = iif([ & ] $ upper(&Contact), [B], [Z])

   A = &&FirstName+[ & ]+left(strtran(&Contact, &FirstName+[ and ], []), at([ ], strtran(&Contact,
       &FirstName+[ and ], [])))
  B = &&FirstName+[ & ]+left(strtran(&Contact, &FirstName+[ & ], []), at([ ], strtran(&Contact,
       &FirstName+[ & ], [])))
  Otherwise = &&FirstName

  Overwrite = 1

[DEPARTMENT]

; Here I am using the Department field to hold any Salutation or Contact name Prefix.
; For instance, if the Contact name were entered as Dr. and Mrs. DJ & Carol Hunt, this instruction
; set would populate the Department field with Dr. and Mrs..

; Pay particular attention to the order of processing in the AutoUpdate section as it is
; critical to the proper processing of this Lookup.ini

  Lookup1 = iif(upper(left(&Contact, 10)) == [MR. & MRS.], [A], [Z])
  Lookup2 = iif(upper(left(&Contact, 8)) == [MR & MRS], [C], [Z])
  Lookup3 = iif(upper(left(&Contact, 4)) == [MRS.], [D], [Z])
  Lookup4 = iif((upper(left(&Contact, 3)) == [MR.] .or. upper(left(&Contact, 3)) == [MRS] .or.
            upper(left(&Contact, 3)) == [MS.]), [E], [Z])
  Lookup5 = iif((upper(left(&Contact, 2)) == [MR] .or. upper(left(&Contact, 2)) == [MS]), [F], [Z])
  Lookup6 = iif(upper(left(&Contact, 10)) == [DR. & MRS.], [A], [Z])
  Lookup7 = iif(upper(left(&Contact, 8)) == [DR & MRS], [C], [Z])
  Lookup8 = iif(upper(left(&Contact, 3)) == [DR.], [E], [Z])
  Lookup9 = iif(upper(left(&Contact, 2)) == [DR], [F], [Z])

  A = &left(&Contact, 10)
  B = &left(&Contact, 12)
  C = &left(&Contact, 8)
  D = &left(&Contact, 4)
  E = &left(&Contact, 3)
```

Note

Remember that each line is a continuous line of code. The lines are wrapped, and indented here for presentation and readability only.

```
  F = &left(&Contact, 2)
  Otherwise = &&Dept

  Overwrite = 1

[LASTNAME]

; Here I am properly populating the LastName field.  This uses the same LastName Instruction
; Set discussed earlier in this chapter.

; Pay particular attention to  the order of processing in the AutoUpdate section as it is
; critical to the proper processing of this Lookup.ini

  Lookup1 = iif(“,” $ Contact, “A”, “Z”)
  Lookup2 = iif((upper(trim(LastName)) == “JR” .or. upper(trim(LastName)) == “JR.”), “B”, “Z”)
  Lookup3 = iif((upper(trim(LastName)) == “SR” .or. upper(trim(LastName)) == “SR.”), “B”, “Z”)
  Lookup4 = iif((upper(trim(LastName)) == “II” .or. upper(trim(LastName)) == “III”), “B”, “Z”)
  Lookup5 = iif((upper(trim(LastName)) == “ESQ” .or. upper(trim(LastName)) == “ESQ.”), “B”, “Z”)
  Lookup6 = iif((upper(trim(LastName)) == “MD” .or. upper(trim(LastName)) == “MD.”), “B”, “Z”)
  Lookup7 = iif((upper(trim(LastName)) == “PHD” .or. upper(trim(LastName)) == “PHD.”), “B”, “Z”)

  A = &alltrim(substr(&Contact, rat(“ “, substr(&Contact, 1, rat(“,”, trim(&Contact))-1))+1,
                rat(“,”, trim(&Contact)) - rat(“ “, substr(&Contact, 1, rat(“,”, trim(&Contact))))-1))
  B = &alltrim(substr(&Contact, rat(“ “, substr(&Contact, 1, rat(“ “, trim(&Contact))-1))+1,
                rat(“ “, trim(&Contact)) - rat(“ “, substr(&Contact, 1, rat(“ “, trim(&Contact))-1))))
  Otherwise = &LastName

  Overwrite = 1

[SECR]

; Here I am using the Secr field to hold Marital Status, and populate it when known.
; For instance, if the Department field contains Mr. & Mrs. the assumption is that they are
; Married.

; Pay particular attention to  the order of processing in the AutoUpdate section as it is
; critical to the proper processing of this Lookup.ini

   Lookup1 = iif((upper(trim(Contact1->Department)) == [MR. & MRS.] .or. upper(trim(Contact1-
                >Department)) == [MR & MRS]), [A], [Z])
   Lookup2 = iif((upper(trim(Contact1->Department)) == [MR. AND MRS.] .or. upper(trim(Contact1-
                >Department)) == [MR AND MRS]), [A], [Z])

  A = Married
  Otherwise = &Contact1->Secr

  Overwrite = 1

[UAADDRESS1]

; Here I am populating an Alternate Address field with the information from Contact1.Address1
; Later this information can be changed if the Contact has, let’s say, a summer address.

  Otherwise = &Address1

[UAADDRESS2]

; Here I am populating an Alternate Address field with the information from Contact1.Address2
; Later this information can be changed if the Contact has, let’s say, a summer address.

  Otherwise = &Address2

[UAADDRESS3]

; Here I am populating an Alternate Address field with the information from Contact1.Address3
; Later this information can be changed if the Contact has, let’s say, a summer address.

  Otherwise = &Address3

[UACITY]

; Here I am populating an Alternate City field with the information from Contact1.City
; Later this information can be changed if the Contact has, let’s say, a summer address.

  Otherwise = &City

[UACOUNTRY]

; Here I am populating an Alternate Country field with the information from Contact1.State
; Later this information can be changed if the Contact has, let’s say, a summer address.
```

Note

Remember that each line is a continuous line of code. The lines are wrapped, and indented here for presentation and readability only.

```
  Lookup1 = iif(trim(&State) $ "MA NY CT VT NH ME RI VA NC TX MI MO KS CA PA FL ND SD HI AL
            KY ID WI DC", "A", "Z")
  Lookup2 = iif(trim(&State) $ "UT NM AZ NV OH IL OR WA OK WV SC DE CO MD GA MN IN NJ IA NE
            TN LA AR MS", "A", "Z")
  Lookup3 = iif(trim(&State) $ "ON AB NS MB QC", "B", "Z")

  A = USA
  B = Canada
  Otherwise = &&Country

[UADEARA]

; Here I am populating an Alternate Dear field with the information from Contact1.Dear
; Later this information can be changed if the Contact has, let's say, a summer address.

  Otherwise = &Dear

[UADEARB]

; Here I am populating a second Alternate Dear field with the information from Contact1.Dear
; Later this information can be changed if the Contact has, let's say, a summer address.

  Otherwise = &Dear

[UALABELA]

; Here I am populating an Alternate Contact field with the information from Contact1.Contact
; Later this information can be changed if you want to have an alternate Contact representation.

  Otherwise = &Contact

[UALABELB]

; Here I am populating another Alternate Contact field with the information from Contact1.Contact
; Later this information can be changed if you want to have an alternate Contact representation.

  Otherwise = &Contact

[UASTATE]

; Here I am populating an Alternate State field with the information from Contact1.State
; Later this information can be changed if the Contact has, let's say, a summer address.

  Otherwise = &State

[UAZIP]

; Here I am populating an Alternate Zip field with the information from Contact1.Zip
; Later this information can be changed if the Contact has, let's say, a summer address.

  Otherwise = &Zip

[uCTaxID]

; Here I am populating a TaxID field for the Client in a proper, and consistent format.
; Let's say that the user enters 061-307-252.  This instruction set would convert that to
; 06-1307252

  Lookup1 = iif(empty(Contact2->uCTaxID), [A], [Z])

  A = &space(0)
  Otherwise = &left(strtran(strtran(Contact2->uCTaxID, [ ], []), [-], []),2)+[ ]+substr(strtran(
              strtran(Contact2->uCTaxID, [ ], []), [-], []),3,11)

  Overwrite = 1

[uMTaxID]

; Here I am populating a TaxID field for the Spouse in a proper and consistent format.
; Let's say that the user enters 061-307-252.  This instruction set would convert that to
; 06-1307252

  Lookup1 = iif(empty(Contact2->uMTaxID), [A], [Z])

  A = &space(0)
  Otherwise = &left(strtran(strtran(Contact2->uMTaxID, [ ], []), [-], []),2)+[-]+substr(strtran(
              strtran(Contact2->uMTaxID, [ ], []), [-], []),3,11)

  Overwrite = 1
```

Note

Remember that each line is a continuous line of code. The lines are wrapped, and indented here for presentation and readability only.

```
[uNNCL]

; Here I am populating a Nick Name field for the Client.  This Instruction Set is using the
; GoldMine macro, &FirstName, to extract the First Name from the Contact1.Contact field.

  Otherwise = &&FirstName

  Overwrite = 1

[uNNMA]

; Here I am populating a Nick Name field for the Spouse or Significant Other.  If the Contact1.Dear ;
field were to contain DJ & Carol, then this Instruction Set would extract Carol from that field.

  Lookup1 = iif([ AND ] $ upper(&Dear), [A], [Z])
  Lookup2 = iif([ & ] $ upper(&Dear), [B], [Z])

  A = &strtran(&Dear, &FirstName+[ and ], [])
  B = &strtran(&Dear, &FirstName+[ & ], [])

  Overwrite = 1

[USERDEF01]

; Here I am populating an Alternate Company field with the information from Contact1.Company

  Otherwise = &Company

  Overwrite = 1

[USERDEF02]

; Here I am populating a Contact field with a true, and consistent format.
; Let's say that the Contact name was entered as DJ & Carol Hunt.  This Instruction Set would
; extract DJ Hunt and place it into this Contact field.

  Lookup1 = iif(([ AND ] $ upper(&Dear) .or. [ & ] $ upper(&Dear)), [A], [Z])

  A = &&FirstName+[ ]+Contact1.LastName
  Otherwise = &&Contact

  Overwrite = 1

[USERDEF03]

; Here I am populating a Spouse or Significant Other field with a true, and consistent format.
; Let's say that the Contact name was entered as DJ & Carol Hunt.  This Instruction Set would
; extract Carol Hunt, and place it into this Spousal field.

  Lookup1 = iif([ AND ] $ upper(&Dear), [A], [Z])
  Lookup2 = iif([ & ] $ upper(&Dear), [B], [Z])

  A = &strtran(&Dear, &FirstName+[ and ], [])+[ ]+&LastName
  B = &strtran(&Dear, &FirstName+[ & ], [])+[ ]+&LastName
  Otherwise = &space(0)

  Overwrite = 1

[uSSNCl]

; Here I am populating a SSN field for the Client in a proper, and consistent format.
; Let's say that the user enters 061366738.  This instruction set would convert that to
; 061-36-6738

  Lookup1 = iif(empty(Contact2->uSSNCl), [A], [Z])

  A = &space(0)
Otherwise=&left(strtran(strtran(Contact2->uSSNCl,[],[]),[-],[]),3)+[-]+substr(strtran(strtran(Contact2-
                >uSSNCl, [ ], []), [-], []),4,2)+[-]+right(strtran(strtran(Contact2->uSSNCl, [ ], []), [-], []),4)

  Overwrite = 1

[uSSNMa]

; Here I am populating a SSN field for the Spouse or Significant Other in a proper, and consistent
; format.  Let's say that the user enters 061366738.  This instruction set would convert that to
; 061-36-6738

  Lookup1 = iif(empty(Contact2->uSSNMa), [A], [Z])
```

Note

Remember that each line is a continuous line of code. The lines are wrapped, and indented here for presentation and readability only.

```
A = &space(0)
Otherwise = &left(strtran(strtran(Contact2->uSSNMa, [ ], []), [-], []),3)+[-]+substr(strtran(strtran(
            Contact2->uSSNMa, [ ], []), [-], []),4,2)+[-]+right(strtran(strtran(Contact2->uSSNMa,
            [ ], []), [-], []),4)

Overwrite = 1
```

I hope, from this example, that you can see how really, and truly powerful the Lookup.ini actual could be. I'm sure that if you can think it, that you can also do it through the use of your Lookup.ini.

GMTray

There has been another tool available for consistently formatting data within GoldMine fields for some years now, and it is called **GMTray**. John Stillman, when he had nothing better to do, designed this little ditty as an example for us developers, but it remains today to be a valuable formatting tool. Additionally, it has the added capability of Starting/Stopping GoldMine at predetermined times on specific days.

Note

GMTray is supplied to you in the eBook download as a zipped file. I would suggest that you unzip these files to a local folder on every Workstations C.:\ drive. Possibly: ***C:\GoldMine\GMTray...***

You should then drag ***C:\GoldMine\GMTray\GMTray\obj\Release\GMTray.exe*** *to the Startup folder on that Workstation.*

Now you can either start GMTray.exe by hand or reboot the Workstation, and it will start automatically.

Unfortunately, there is no common shared folder, and GMTray will need to be configured the same for each Workstation to enjoy corporate consistency.

Possibly you have GMTray set up, and working from your Workstations. If so, it is time to configure GMTray by right clicking on the GoldMine treasure chest in the Workstation System Tray (lower right-hand corner of Windows), and selecting **Configure** from the local menu which will bring up the screen shot shown here in Figure 6- 2.

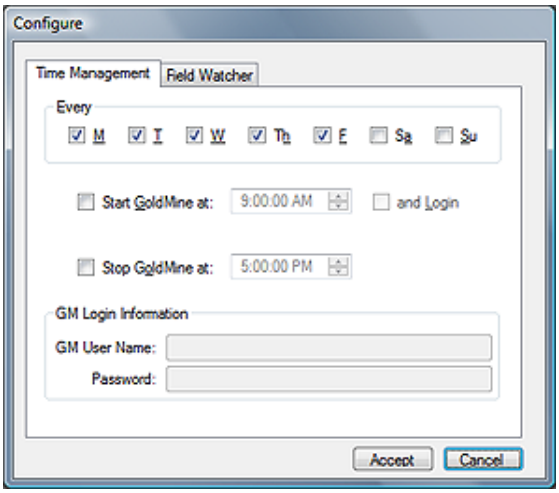


Figure 6-2

The default tab is **Time Management**, and it is from here that one may schedule the starting and stopping of GoldMine for Workstations that are always left running. This was particularly important in the old dBase versions of GoldMine, however, it remains a nice feature even for today's GoldMine. Why leave applications running unnecessarily? The first frame, **Every**, contains simple checkboxes for the days of the week that you wish to have GMTray Start/Stop the Workstations GoldMine. Simply check those days that you wish GMTray Start/Stop GoldMine, and uncheck those days where you do not wish any activity from GMTray.

Naturally, no matter how many days that you have selected in the **Every** frame, nothing will happen unless you have completed the rest of the information in the **Time Management** tab. The first option that is of concern to us is whether to ☐ **Start GoldMine at: 9:00:00 AM**. Obviously, this is the time at which you wish to have GMTray run your GoldMine application from the Workstation. Selecting this alone will cause GoldMine to start up to the splash screen, and sit there waiting for a user to login. Alternatively, they may choose the option ☐ **and Login** which will utilize the **GM Login Information** supplied in that frame to login to GoldMine directly bypassing the splash screen.

Well, once GoldMine is started, and on Workstations that are left running continuously, one may wish to shut down the application as well. To do so, one would have to select to ☐ **Stop GoldMine at: 5:00:00 PM**. For this book, I have chosen to use the default Start/Stop times, however, once selected these times become enabled and the user may address them as is appropriate to their schedule.

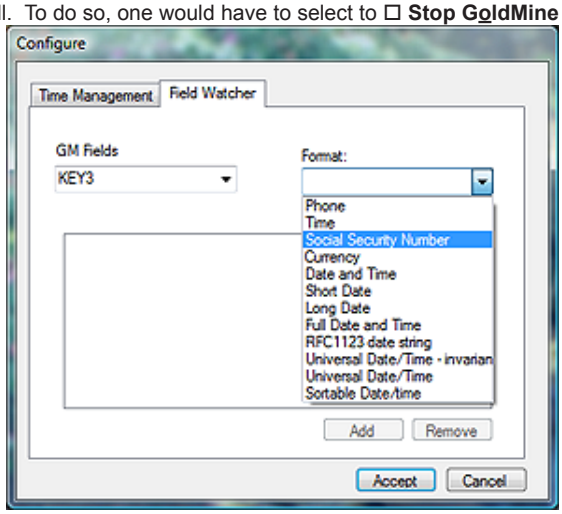


Figure 6-3

Lastly, on the **Time Management** tab, if one wishes GMTray to log the Workstation directly into GoldMine, one must supply the **GM User Name:** and the **Password:** to be utilized for the login in the **GM Login Information** frame.

Next, on the **Field Watcher** tab, we have the ability to set up GMTray to watch certain fields within GoldMine for a change, and when changed, to format the field as specified. One would select field to be watched under **GM Fields**,

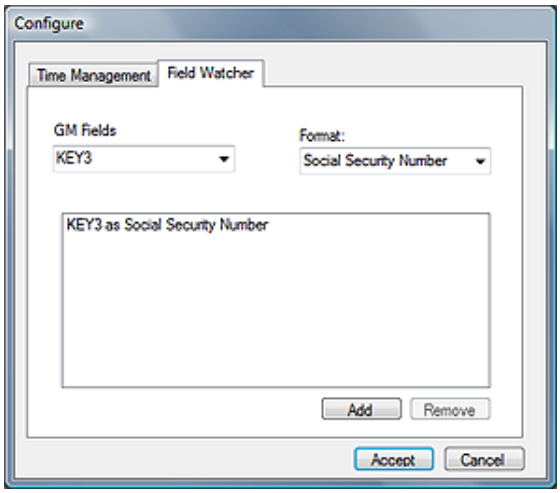


Figure 6-4

and the formatting to take place under the **Format:** field. Once set, one would click on the **Add** button to send it to the list of watched fields.

As you can see here in Figure 6-4, I have set my **Key3** field format as a **Social Security Number** format. Hence, with GMTray running in the System Tray, upon entering **AB123456789** into the Key3 field, GMTray would, and did, convert the value to **123-45-6789**.

As shown in Figure 6-3 on the previous page, there are many other formatting conditions that can be assigned to any field. GMTray eliminates the need for using the Lookup.ini to simply format fields, and, as a bonus, allows you to Start & Stop GoldMine automatically on predetermined days and at predetermined times. For a tool that John developed to show off the API functionality to developers, this is actually a very practical end user tool, and, best of all, its free.

7

In This Chapter

This is another one of those fun chapters for me to write, and in this writing I'm going to incorporate the **Relationship Tree** which, although it has been around for a while, I have yet to really discuss in any of my previous books. Prior to GoldMine Premium, the **Relationship Tree** was known as the **Organization Tree**, but we'll get into more of that later in this chapter.

How useful is the data if you can not easily access that data, or find information contained within the database? It's not very useful at all. Users must be able to cluster the data for mail merges, fax blasts, e-mail blasts, by products purchased, or by any number of various attributes, and GoldMine has five separate ways to perform these sorts of activities.

In this chapter, I am going to introduce you to **Filters**, **Preview**, **SQL Queries**, **Groups**, **Record Tagging** and the **Relationship Tree**, in that order. I say, in that order, as this order is not the same one as is presented on Filters and Groups dialog form shown below in Figure 7-1. I am intentionally presenting these to you in this order as **Groups** could, as well, be built based upon a **SQL Query**, and I thought that it would be best if I covered the **SQL Query** ahead of **Groups** for that very reason.

Note

*The **View Filters:** field is no longer sticky in GoldMine Premium. This means when the user closes the **Filters and Groups** dialog form, the selected UserID will no longer be saved. The next time this user enters the **Filters and Groups** dialog form, they will enter it displaying the filters of the logged in UserID.*

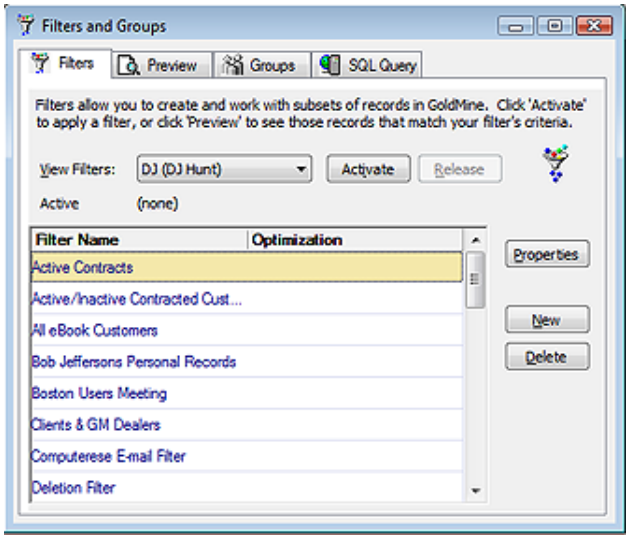


Figure 7-1

What are the distinguishing characteristics or the three methods? Well, **Filters** and **Groups** can be activated, and used, once created, for mail merges, fax blasts, etcetera. On the other hand, the **SQL Query** only pulls data from the different tables into a single query table. The resulting table is normally referred to as a cursor. You can not activate the results of the **SQL Query** and, in turn, use it as you would a **Filter** or a **Group**. You can, as I stated earlier, build a group of records based on the AccountNos in the cursor, but, and more importantly, you could select **Output to ► Microsoft Word**, **Microsoft Excel** or to the **Clipboard** directly from within the cursor. I would point out that there is a similar **Output to ►** option for **Groups**, but it doesn't put out anywhere near the information that can be gathered through a single SQL Query. In fact, this option, from within **Groups**, only sends the **Member**, **Sort**, and **Reference** information to Word or Excel. Therefore, in short, **Filters** and **Groups** point to the contact records that contain the information that you are seeking, where as the **SQL Query** result cursor actually contains the information itself, and not just pointers to records.

What is the difference between a GoldMine **Filter** and a GoldMine **Group**? An excellent question, indeed. They both display records based on some predefined criterion. The filtered information is, however, **Dynamic** while the grouped information is **Static**. Any time a **Filter** is activated against a da-

tabase, all of the records that meet the filter condition(s) will be available for various activities that employ filtered sets. On the other hand, **Groups** are a fixed list of records that are created at the time that the individual Group is built. The pointers to the records included in the Group are stored in the **ContGrps** table. This is more of a snapshot in time of which records met your conditions at the time that the Group was created. Yes, one can add to a Group after its has been built, changing that particular Groups point in time. One must remember to perform that step prior to deploying the Group in activities that work with subsets of records. **Filters - Dynamic** versus **Groups - Static**, quite a significant difference in result sets wouldn't you say?

There is another significant difference between **Filters** and **Groups**. Using the GUI, filters can only be built against information contained in the **Contact1** and **Contact2** tables. A group, however, may be generated based on filtered records, previewed records, SQL queried records, scheduled calendar activities, complete history activities, supplemental contact data, tagged records or search results. Wow! Static, though they are, one could build a list of accounts based on virtually any table in the GoldMine database that contains the **AccountNo** field.

Filters

As I stated, I will begin with the Filters tab. I like to envision a filter as a sieve. One selects the size of the screen on the sieve (the condition), and when the sieve is filled with grains of sand, only those that meet the sieve screen size pass through. All other grains of sand are still in the sieve, however, one can only work with the material on the ground that has made it past the sieve screen.

Let's discuss the basic use of the filter GUI first, and then we can continue on into some more advanced material. Referring to Figure 7-1 on the previous page, the first field that one encounters is the **View Filters:** field. Here, if the user has the proper rights, is a drop-down list of all of the users that GoldMine has within its system, as well as the one group **(public)**. Filters are created for a specific UserID or for the (public) user category. Any user, having proper access rights, may employ any filter constructed by any other user. In as much, it is important, when naming a filter, that the creator be cognizant of the fact that other users must be able to distinguish from the **Filter Name** exactly what the filter will accomplish.

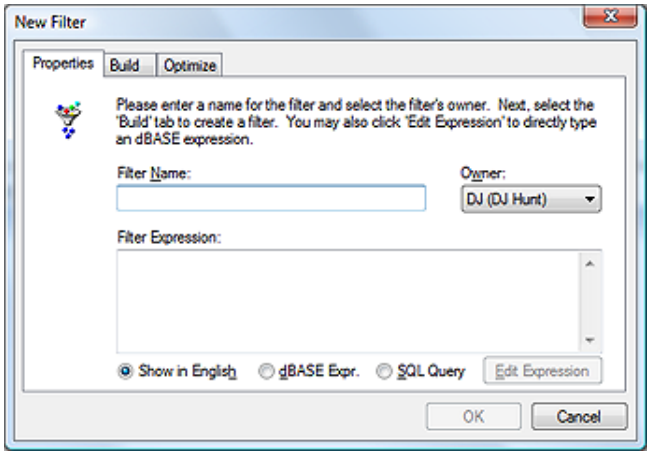


Figure 7-2

Now, click on the **New** button to bring up the **New Filter** dialog form as shown here in Figure 7-2. The first field that is available, is the **Filter Name:** field. This is the field where the user is asked to supply as descriptive a representation of the filter as the allotted space permits. For this example, I would insert into this field, **Created On > 8/1/2009 - CreateOn Field**. I will leave the **Owner:** field set at **DJ (DJ Hunt)**, and, at this point in time, the **Filter Expression:** field will remain empty until we have finished building this new filter.

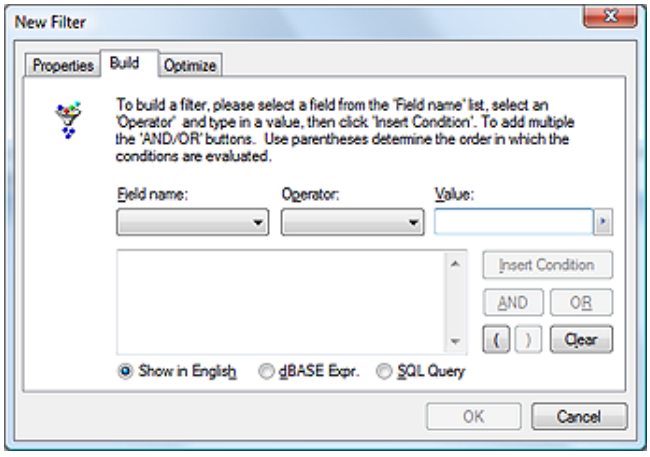


Figure 7-3

Click on the **Build** tab to bring up the dialog form shown here in Figure 7-3. This is the GUI that is utilized to build the filtering condition(s). There are three input fields, the first two are drop-down list, while the third is a text based field. The first list field contains the **Field name:** list. Clicking on the drop arrow to view the list, the user should notice that only those fields that are contained in their **Contact1** and **Contact2** tables appear in this list. For example, the user will not see e-mail address in this list, as the e-mail address is contained in the **ContSupp** table. For this example, I am selecting the **CreateOn** field from the drop list.

Note

*I've been meaning to ask FrontRange why they maintain the use of the O **dBase Expr.** radio button in the **New Filter** dialog form in GoldMine Premium.*

In the next field, the **Operator:** field, I will select **Greater or Equal** from its drop list. The user will notice that the operators are in plain English. Here is a list of the available operators in this drop list:

Equal to
Not Equal to
Greater than
Less than
Greater or Equal
Lesser or Equal
Begins with
Contains
Does Not Contain
Is Empty
Is Not Empty

In the **Value:** field, I have entered **8/1/2009**, and at this point, I would click on the **Insert Condition** button which will append to the memo field, in plain English, **Createon is Greater or Equal "8/1/2009"**. I could add more to this expression using the **.and.** or the **.or.** option, but for this exercise let's just click on the **OK** button to save the filter.

In my 4607 record database, if I right-click on this filter to bring up the local menu, and if I select **Count...** from the local menu, the filter returns a result of **Count: 8 (0.2%)**. This count is returned in milliseconds without having employed the **Optimization** feature for the filter. This filter may not work well against older GoldMine installations, however, as the **Contact1.CreateOn** field is a relatively new field to the GoldMine tables. Old time GoldMine users may have a number of records with no data in the **Contact1.CreateOn** field.

Up until recent releases, GoldMine had always derived the create on date by analyzing the **AccountNo** field. After FrontRange incorporated the **Contact1.CreateOn** field into GoldMine, they needed to come up with something that would allow older upgrading users to access the created on date information. The result was the creation of the **accdate()** function (Refer to Appendix A). This function can extrapolate the creation date using the **Contact1.AccountNo** field, however, unlike previous versions of GoldMine, in GoldMine Premium you cannot create a dBase expression unless you can create an equivalent SQL where clause. Hence, based on our knowledge of the **Contact1.AccountNo** field (refer to Chapter 8 - The Tables), we know that the first 6 characters of the **Contact1.AccountNo** field are the create on date.

Click on the **New** button again. This time, let's give a **Filter Name:** of **Created On > 8/1/2009 Using AccountNo**. Now, instead of clicking on the **Build** tab, select the O **dBase Expr.** radio button, and then click on the **Edit Expression** button. This should bring you into the **Edit Expression** dialog form where you are free to type whatever legal dBase expression you desire. I'll show you other uses for this later.

Here is the expression that I would like you to type into this field:

```
left(AccountNo, 6) >= [A90801]
```

Now, instead of clicking on the **Build** tab, select the O **SQL Query** radio button, and then click on the **Edit Expression** button. This should bring you into the **Edit Expression** dialog form where you are free to type whatever legal SQL where clause that you desire.

Here is the where clause as I have it entered:

```
left(C1.AccountNo, 6) >= 'A90801'
```

You may have noticed, although I did not type the **WHERE** portion of the clause, that it still appears in the dialog form, Figure 7-4. The dBase & SQL expressions equate, hence, the **OK** button is enabled, and I can save this filter. Best of all, it too will function as expected. This, however, is not a very dynamic filter as it must be modified each time that you plan to place it in use. I think that I prefer the old ways where you could put in a dBase expression or a where clause, and it would evaluate properly.

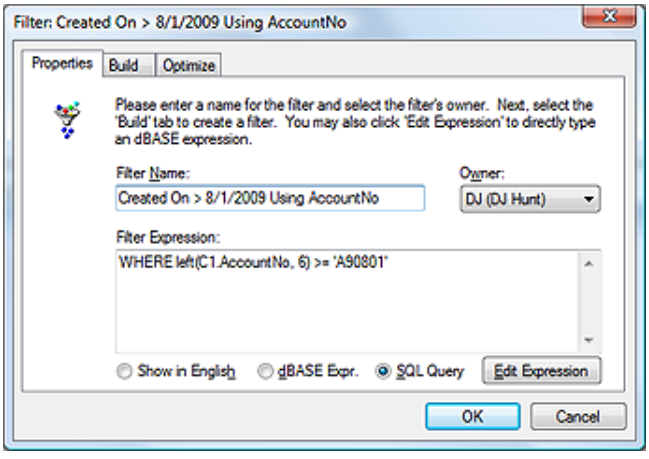


Figure 7-4

Tip

*In past versions of GoldMine, particularly the GoldMine Standard Edition, I would have employed the **Optimize** tab, and its ability to speed up filters by using existing GoldMine Indexes. To date, I have found that using this feature in GoldMine Premium has no effect on positive performance, and, conversely, sometimes causes the filter to not function properly.*

The reader is advised to test this feature against the version of GoldMine that they are using to determine whether its usage is beneficial to their needs.

*The reader is further advised, if they wish to Optimize their expressions, that they do so by modifying the dBase and SQL expressions utilizing the indexed mirror fields (refer to Chapter 8 - The Tables) when possible. All of the indexed mirror fields begin with **U_** such as:*

Company mirrored to U_Company

Note

*I've been meaning to ask FrontRange why they maintain the use of the O **dBase Expr.** radio button in the **New Filter** dialog form in GoldMine Premium.*

Where we could have created more dynamic filters before, we are more hampered in GoldMine Premium unless you are extremely fluent in SQL Query development.

Once completed, click on the **OK** button of the **Edit Expression** dialog form, and then on the **OK** button of the **New Filter** dialog form. Now you should be back to the **Filters and Groups** dialog form, right-click on the new filter and select **Count...** from the local menu. Against my database the results were **Count: 12 (0.3%)**, and there was no discernible difference in process speed.

In previous books I was able to address this question from the forum: *"How can I build a filter for all of my contacts that have an e-mail address?"* Alas, GoldMine Premium has taken away this ability to the best of my knowledge. No longer can you utilize:

```
.not. empty(&EmailAddress)

.not. empty(Fax) .and. empty(&EmailAddress)

upper(c1.Phone1) > ' '
  and c1.AccountNo not in
    (select AccountNo
    from ContHist
    where (OnDate+30) >= getdate())
```

That is, unless you can devise a SQL where clause, or a dBase expression, to match the converse of these items. So let me now show you how this needs to be done in todays GoldMine Premium.

Let's start by creating a **New** Filter, and we'll call it: **Contacts with E-mail Address**. Good. Now let's click on the radio button for ◉ **dBase Expr.**, and then on the **Edit Expression** button. In the **Edit Expression** dialog form, enter:

```
.not. empty(&EmailAddress)
```

You remember that dBase expression, don't you? Go ahead and click on the **OK** button. No change yet from the old days, however, we now need to balance the equation with a comparable SQL Where clause. Let's select the radio button ◉ **SQL Query**, and then on the **Edit Expression** button again. This time, however, in the **Edit Expression** dialog form, we'll enter:

```
C1.AccountNo in
  (select AccountNo
   from ContSupp
   where U_CONTACT = 'E-MAIL ADDRESS'
     and Zip like '_1%')
```

Now let's click on the **OK** button on the **Edit Expression** dialog form, and then again on the **OK** button on the **Filter:** dialog form. That's it, we're done. Right-click on the Filter and select **Count...** from the local menu. My count shows: **Count: 3873 (84.0%)**, and took less than a second to generate that number.

Here are the expressions for the converse of the above filter, Contacts without a Primary E-mail Address:

```
empty(&EmailAddress)

C1.AccountNo not in
  (select AccountNo
   from ContSupp
   where U_CONTACT = 'E-MAIL ADDRESS'
     and Zip like '_1%')
```

Note

You will not be able to enter Carriage Returns or Line Feeds, and the where clause is shown here formatted for presentation purposes only. Your where clause will be in one continuos line.

Preview

My Bad! My editor picked this up. I had stated on the first page that I wanted to include two new sections in this chapter for this book. One was the **Preview** section, while the other was the **Relationship Tree** section. Low and behold, I forgot both sections, hence, the Editors rejection of the manuscript.

Looking back at Figure 7-1, you may have noticed the **Preview** tab in between the **Filters** tab and the **SQL Query** tab. It is here that FrontRange has chosen to enhance the GoldMine Filters. Before we leave the **Filters**, however, let's create two rather simplistic filters to use in this exercise.

The first to select all of my Clients for which I maintain a marker in the **Contact1.Key1** field labeled **Category**.

```
Category (Key1) Contains Client
```

Now I'm going to create another filter where:

State Equal to MA

As that filter was the last filter that I created, it remains highlighted (selected). While this filter is highlighted, let's choose the **Preview** tab, refer to Figure 7-5. In the upper box you'll notice our filter ⦿ **In Englis**h as opposed to being **In dbase**, and it is:

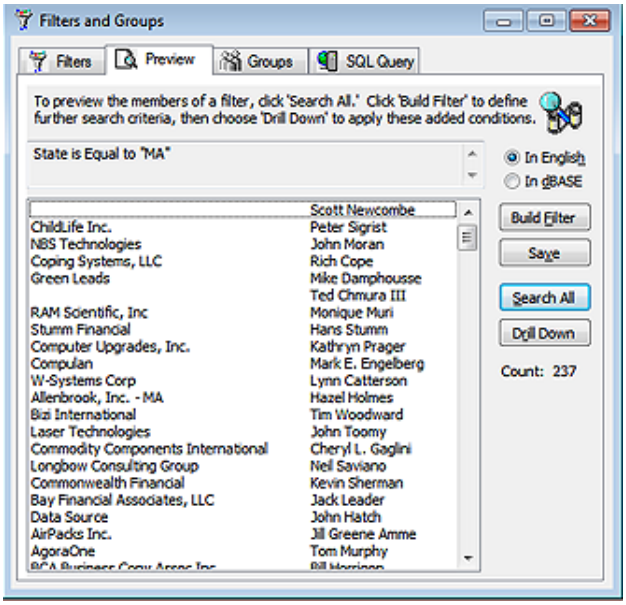


Figure 7-5

State is Equal to "MA"

Normally, the second box would have been blank at this point, however, for Figure 7-5, I had already clicked up the **Search All** button which delivered 237 records.

Now let's go back to the **Filters** tab, and highlight the **All Clients** filter. Immediately thereafter, switch back to the **Preview** tab, and look at the top box this time. You'll notice that it now contains this filter expression in English.

Category (Key1) Contains "Client"

However, if I click upon the **Drill Down** button, and I did for Figure 7-6, you'll notice that the expression has changed to:

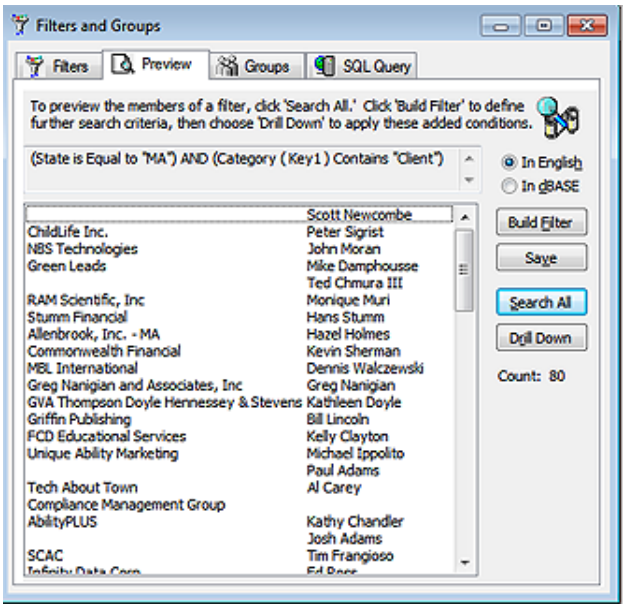


Figure 7-6

(State is Equal to "MA") AND (Category (Key1) Contains "Client")

On top of that you'll want to look at the new **Count: 80** figure. Now I know that this simple filter could have been built from the **Filter** tab, however, take this to the level of a couple of complicated filters, and you have a nice system. Especially when you realize that you can now click upon the **Save** button to update the information in the highlighted filter on the **Filters** tab, or to create a totally new filter utilizing the expression now created.

Alternatively, you could modify the expression right from the **Preview** tab, by clicking on the **Build Filter** button. WOW! Can it get any easier?

SQL Queries

I consider the SQL Query to be the most under utilized gem contained within the GoldMine application. Think about it, people have paid me $249.00 for my Beyond Gold application (no longer available) to export the other **Contacts** from their GoldMine **ContSupp** table when they could have just as easily utilized the **SQL Query** tool in GoldMine. Individuals get fixated on the fact that GoldMine can not export anything except information from the **Contact1** and **Contact2** tables unless exporting to GoldMines own interpretation of XML. They go to the forums and ask, often, *"I have to send a list of other contacts to a mailing house, how can I export this information?"*. Then you see all of the third party solution providers hawking their wares, as I recently saw it stated, and forgetting to even mention that this could just as easily be accomplished within the GoldMine product itself.

By the way, it is my personal policy, when answering questions in the forums, to only give the GoldMine solution answer to the question unless there is no GoldMine solution. Then, and only then, do

I mention the possibility of trying a third party product. Have you noticed that this is a particular sore spot for me?

All right, let's move on then. As a prerequisite to something that I am going to attempt, I would ask you to go to any record, that is in the State of California or Georgia. As an additional requirement, you want to make sure that the record has Additional Contacts. Double-click on any of the Additional Contacts to bring up the **Additional Contact at ...** dialog form. Place an **X** in the ☑ **Merge:** field, and then click on the **OK** button. We are going to want to use that later.

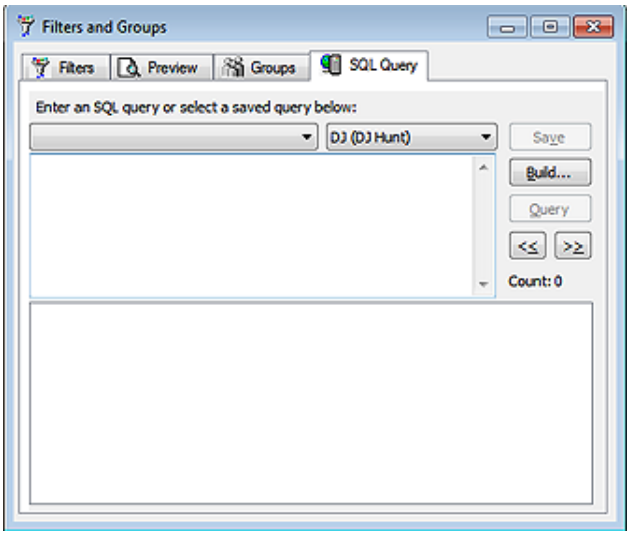


Figure 7-7

Good, now that this prerequisite is established, let's start out by saying that you want to send a list of **Additional Contacts**, that are linked to a primary contact record, and where the primary record resides in the state of California or the state of Georgia, to a mailing house. An awkward and possibly unrealistic request I know, but this will serve as the basis for my first example.

Looking here at Figure 7-7, in the medium size box just under the user name, let's type the following:

```
select CS.*
from Contact1  C1,
  ContSupp  CS
where C1.AccountNo = CS.AccountNo
  and C1.State in ('CA', 'GA')
  and CS.RecType = 'C'
```

Clicking on the **Query** button against my dataset pulled **238** records (we use to be told the seconds that had evolved, however, in GoldMine Premium this is no longer the case). Just so that you know, my dataset currently has 4,613 records (I cleaned up my database since my last book), so 238 records pulled from that in less then a second is not too shabby. By the way, this and any SQL select statement, can not contain any carriage returns or line feeds. The select statement above contains line feeds, carriage returns, and spaces. As of this writing, GoldMine is very forgiving, and should you accidently include any of these in your select statement, GoldMine is currently ignoring their inclusion.

So let's talk about this statement a little. To keep my typing to a minimum, I used what is called aliasing of the table names. Look at the **from** clause in the statement above to see how I accomplished this. I stated the actual table name trailed by a space, and then the alias name for the table as, **Contact1 C1**. I am working against two tables, so I separated them in the **from** clause of the statement by a comma (**,**). Now that I have aliased the table names, I can use the alias everywhere else instead of typing out the full table name each time.

My select statement itself, states, display all fields from the **CS** (**ContSupp**) table. I used the wild card, the asterisk (*****) to accomplish this. I then proceeded to set up some conditions for pulling the information. Conditions, haven't I heard that someplace before? Another name for a condition is a filter. Yes, I am filtering our tables using conditions. The first condition is what is called a simple join clause. I am saying to only look at records where there is a match between the **C1.AccountNo** field and the **CS.AccountNo** field. Remember that the **AccountNo** field is the primary relationship key between most of the database tables within GoldMine.

Now that I have joined the two tables, effectively making them one big table, I can apply additional conditions. In this example, you'll remember, I only wanted the other contacts if the primary contact was from the state of California or Georgia. Therefore, my next condition and **C1.State in ('CA', 'GA')**. Now both conditions, the join and this statement, must return a **True**, based on the Boolean **and**, before the fields will be displayed in the query result window.

Lastly, I needed to add one more condition. The **ContSupp** table contains many records of varying types that are linked to a single contact record. As I am only looking for **Additional Contacts** to the primary record, I am not interested in any **Detail** records, **Linked Documents**, or **Referrals** to name a few of the possible records that could be pulled from the **ContSupp** table. Records in the **ContSupp** table are differentiated by type in the RecType field.

WARNING

As a general rule, you should not copy and paste from this book as things like "Variable" have styles associated with them, and we often use line wrapping for better book presentation. Most of this information will not translate well when pasted elsewhere.

All SQL Queries discussed in this book are included in a NotePad created document from which you may copy and paste directly into GoldMine.

Note

*Refer to **Chapter 8**, **The Tables** for a complete understanding of the **ContSupp** table, and it's various **RecType** values.*

The following values are possible type values contained in the **RecType** field:

A **Record Alerts**
C **Additional contact record**
E **Automated Process attached Event**
H **Extended detail header**
L **Linked document**
O **Relationship tree**
P **Detail record (formerly known as a Profile record)**
R **Referral record**

As I am only interested in additional contacts to the primary record, I segregated that set using the and **CS.RecType = 'C'** clause in my statement. For now I ask you to click on the **Save** button, and save this query. This is a very nice feature, unlike the **Groups**, as you will see later, which you have to rebuild from scratch each and every time, GoldMine allows you to save the SQL Query to be utilized again and again as often as you want. I would remind you, when naming your query that you should name it such that, a year from now when you look at that name, you will know from the name exactly what the query does. As always, you are limited in the number of characters, **40** in this case, when naming a query, so use them wisely.

After you have saved the query, let's try to run it against your database by clicking on the **Query** button. Against my database of **37,520** ContSupp records, this query pulled **238** records.

Now I want to change the requirements a little. Everything as before applies, except that I don't want to show all of the fields in the **ContSupp** table. I only want to display the fields necessary to do a mailing. Additionally, I want to make certain that the records in question have an address. What good is a mail list without addresses? This, then, would cause our select clause in the statement to change to the following:

```
select CS.Contact,
 CS.Title,
 CS.Address1,
 CS.Address2,
 CS.Address3,
 CS.City,
 CS.State,
 CS.Zip,
 CS.AccountNo
from Contact1 C1, ContSupp CS
```

WARNING

*Even though **CS.AccountNo** is not required by the mailing house, it is required so as to bind the **Join** properly.*

This, then, would cause our where clause in the statement to change to the following:

```
where C1.AccountNo = CS.AccountNo
 and C1.State in ('CA', 'GA')
 and CS.RecType = 'C'
 and CS.Address1 is not null
  and CS.Address1 > ''
```

Let's save this again, but this time, select to update the current query. Now run this query, and you should have culled your query results down. Against my database of **37,520 ContSupp** records, this query pulled **238** records again. I told you that I cleaned up my database before journeying into this book.

Well let's cull it down even further. This is a Christmas Specials mailing that I am sending to my clients through the mailing house. I only want to send this mailing to those that are designated in our database as wanting to receive this type of mailing.

Let's try this now:

```
select CS.Contact,
  CS.Title,
  CS.Address1,
  CS.Address2,
  CS.Address3,
  CS.City,
  CS.State,
  CS.Zip,
  CS.AccountNo
from Contact1 C1, ContSupp CS
where C1.AccountNo = CS.AccountNo
  and C1.State in ('CA', 'GA')
  and CS.RecType = 'C'
  and CS.Address1 is not null
  and CS.Address1 > ''
  and CS.MergeCodes like '%X%'
```

Did I throw you a curve there? I hope not. As there could be many codes in the **20** character **MergeCodes** field, I only wanted the record if one of those merge codes was an **X**. To do that I put in the wild card percent (**%**). By putting one before, and after the **X**, I am stating that I don't care if there is anything to the left of the **X**, and I also don't care if there is anything to the right the **X**. I just want to know if there is an **X** contained anywhere in the **MergeCodes** field.

Running this query, if you followed our prerequisite setup, should result in one record which, by coincidence is how many records I happened to pull. Probably not worth using the mailing house for this few? For this book, however, I will send the name over. The mailing house doesn't have GoldMine, and I can't export this, as the information is contained in the **ContSupp** table. How am I going to get this over to the mail house? The answer is simple. I can get this information over to them in any format that they can handle. I am just going to do it by going through Microsoft Excel first. Obviously, I make the rash assumption that you have Microsoft Excel installed on your computer.

WARNING

*If there is a **Note** (**Image**) type field in your query result, and if that note contains carriage returns, line feeds or both, then the **Output to ► Excel...** may not be a viable option. **CR** and **LF** in a note, when sent to Excel, could produce some unexpected, and difficult to handle data representations.*

Over your query result table (the cursor), right-click anywhere and select **Output to ► Excel...**, and watch what happens. Excel should start, and the query table with the field names should have been populated on an Excel spreadsheet. Everything should be highlighted, so the user would immediately want to go to the Excel menu and select **Format ► AutoFit Column Width** selection in Excel 2007. That will adjust the column widths to show all of the data properly. Now it is simply a matter of selecting the **Save As...**, and then saving the spreadsheet as any file format that your mailing house can handle. The dBase option is no longer available as of Office 2007, however there are many other file types such as .xls, .csv, and more file to which you could save this file. You could then send that file on to the mailing house for their processing as an attachment to an e-mail message. Slick, quick, and you never left your chair.

Now, is there really any need to go out, and to purchase those third party tools to enable exporting from areas in GoldMine that are not part of the **Contact1**/**Contact2** tables? I think not. I think that you can both save your money, and you can have lots of fun designing your own queries to pull data that your organization can use in various formats. For the rest of this section in this chapter, I will just give you a few queries that are most often requested, and which may be useful to your organization as well. Even if they, themselves, are not useful they may contain elements that will help you in your own design of a **SQL Query**.

I don't know why, but some bug in GoldMine can cause contact records to be assigned more than one Primary E-mail Address. Obviously this would not be good when doing blast e-mails to clients. Here is a select statement that will display the account number, and the number of designated Primary E-mail Addresses for each contact record that has more than one designated Primary E-mail Address account. After running this select, should you have any results, you must take care to assign only one Primary E-mail Address for the specified record.

Note

You must be vigilant in your database maintenance. I just ran this query to test it for the book, and found two more records in this state in my system.

■ **Contacts - with > 1 Designated Primary E-mail Address**

```
select distinct AccountNo,
  count(*)
from ContSupp
where RecType = 'P'
  and Contact = 'E-mail Address'
  and Zip like '_1%'
  group by AccountNo
  having count(*) > 1
```

I would like to point out the number of underscores as this is very important, and there is only one. There is one underscore before the **1** as a single charater wild card. The **%** is a generic wild card.

```
and zip like '_1%'
```

The underscore acts as a single character wild card. So in this case, I am stating that I don't care what is in positions 1, 3 & 4 of the **ContSupp.Zip** field, but that the second character must contain a 1. This is another form of wild card usage when building your SQL Queries, and you should keep it in mind as it does come in handy when you are looking for a specific character at a specific position. You could, of course, also have used the **substring()** function to cull out the right character(s).

WARNING

In earlier versions of GoldMine Premium, I have noticed that this query does not display the Primary E-mail Address in all cases on the first run. I have found it necessary, for whatever reason, to refresh the Primary E-mail Address switch for the records, and from then on the query has functioned as expected.

Now that you have found, and corrected all of those records that had multiple Primary E-mail Addresses, you may want to do an e-mail blast mailing to all of those contact records that have a Primary E-mail Address.

■ **Contacts w/Primary E-mail Address**

```
select distinct
  C1.AccountNo,
  C1.Company,
  C1.Contact,
```

```
  CS.ContSupRef+CS.Address1 as 'E-mail'
from Contact1 C1,
  ContSupp CS
where C1.AccountNo=CS.AccountNo
  and CS.Contact='E-mail Address'
  and CS.Zip like '_1%'
```

Here we go with the reverse situation. I need to know all of those records that do not have a Primary E-mail Address. Who knows, maybe you want to mail them a merge letter (how passé) instead. Well that select statement is a slight modification of the one above.

■ **Contacts w/o Primary E-mail Address**

```
select AccountNo,
  Company,
  Contact,
  Address1,
  Address2,
  City,
  State,
  Zip
from Contact1
where AccountNo not in
 (select C1.AccountNo
  from Contact1 C1,
    ContSupp CS
  where C1.AccountNo=CS.AccountNo
    and CS.Contact='E-mail Address'
    and CS.Zip like '_1%')
```

There you go. All the names in your database that do not have a Primary E-mail Address, and all packaged together nice and neat, ready for your mailing house to process. Well this is really nifty isn't it? I am doing a double select statement again, and as before, I am locating all of those that have a Primary E-mail Address. Notice the not in piece of the statement. If I have a list of those that have a Primary E-mail Address, comparing it to our list of all contacts, then those that are not in the first list but are in the second list must be those without a Primary E-mail Address. Maybe a bit confusing, but after you get a few of these under your belt it'll be natural to you.

Well then, you are in GoldMine Premium Edition so why not combine all of the information from your Primary Contacts and Additional Contacts?

■ **Contacts & Add Contacts w/E-mail**

WARNING

I have tested all of these select statements by Copying & Pasting them from this book into my GoldMine Premium Edition of GoldMine, and they have all worked as expected, except where specifically noted.

*You should do the same, and not attempt to retype them, however, you **must** realize that the single and double quotes are formatted, and must be coverted to plain text in the SQL Query analyzer before testing the query.*

```
select AccountNo,
  Company,
  Contact,
  Address1,
  Address2,
  City,
  State,
  Zip,
  Phone1,
  (select top 1 ContSupRef+Address1
    from ContSupp CS
    where CS.AccountNo = C1.AccountNo
     and Contact = 'E-mail Address'
     and Zip like '_1%'
     ) as 'E-mail'
from Contact1 as C1
union select CS.AccountNo,
  CS.Address3 as Company,
  CS.Contact,
  CS.Address1,
  CS.Address2,
  CS.City,
  CS.State,
  CS.Zip,
  CS.Phone,
  CS2.ContSupRef+CS2.Address1 AS Email
from Contact1 C1,
  ContSupp CS,
  ContSupp CS2
where C1.AccountNo=CS.Accountno
  and CS.RecID = CS2.LinkAcct
  and CS.RecType='C'
order by C1.Company,
  C1.Contact
```

Yup, I threw a **union** clause in there to achieve my goal, and, because of the rule that you must have as many fields in the union table as are in the select table, I chose CS.Address3 (usually blank) to fill the slot in the union side where there is actually no Company field. So we have now taken all of the Contact information from our database, and placed it into a single cursor from where we can output it to Excel or Word. This is an extremely handy query.

In my next select statement I want to show you another statement that is commonly requested. *"I would like to see all of the records in my database for which there has been no History over a defined period of time."* In my example, I want those that have had no history within the last 30 days. Here is an example of the select statement for GoldMine Premium that fulfills this request:

WARNING

*GoldMine always likes to see the **AccountNo** field in a query so that GoldMine can synchronize to the Contact record from the resulting cursor. If GoldMine doesn't see the **AccountNo** in your select statement, GoldMine will add it automatically. In a union select this will cause problems unless you add the **AccountNo** to both sides of the union statement.*

■ **Contacts w/o History in Last 30 Days**

```
select Contact,
  Company,
  AccountNo
from Contact1
where AccountNo not in
 (select AccountNo
  from ContHist
  where OnDate + 30 >= getdate())
order by Company,
  Contact
```

Naturally, one might also be interested in those records that have had no history in the last 30 days, and that are not scheduled for anything currently, or in the future. Getting complicated, huh? Let's see if we can compound this.

■ **Contacts Not Scheduled, and without History in Last 30 Days**

```
select Contact,
  Company,
  AccountNo
from Contact1
where AccountNo not in
 (select AccountNo
  from ContHist
  where OnDate + 30 >= getdate())
 and AccountNo not in
   (select AccountNo
    from Cal)
order by Company,
  Contact
```

Where the Contacts w/o History in the Last 30 Days pulled 7,498 records from my database, Contacts Not Scheduled w/o History in Last 30 Days pulled 7,450 records.

All right now, here are two more simple ones. In the older dBase versions of GoldMine, one had an idea of how many records were in their Contact1 table by looking at the **Summary** tab **Record: 2 of 19**. With GoldMine Premium, and with all previous Corporate Edition installations of GoldMine, the same does not hold true. A user can, however, acquire a count against any table with this simple select statement:

■ **Count - Records in a Table**

```
select count(*)
from Contact1
```

Well, what were you expecting? I did say it was simple. Let's expand on that just a little. How many user defined fields have you created in your GoldMine? Don't know? Don't feel like counting them one by one in the **User Defined Fields** dialog form? Try this select statement:

■ **Count - User Defined Fields**

```
select count(*)
from ContUDef
where Field_Name like 'U%'
```

In previous releases of the Hacker's Guide series of books, this was where I ended my discussions of SQL Queries, however, as I have learned so much more about SQL Queries since my prior writings, I thought that I would stick a few more queries into this section of this chapter. I needed to develop a query for a client that I thought would be useful for you, my readers. It involves the use of the union clause of the select statement.

Scenario: I wanted to select all of the companies for which the company name begins with Computer. In addition, I wanted their Primary Contacts, and all of their Secondary Contacts. Here is the select statement against GoldMine Premium to produce this result set:

■ **Contact/Additional Contacts - from Computer Companies**

```
select Company,
  Contact,
  AccountNo
from Contact1
where Company like 'Computer%'
Union select Contact1.Company,
  ContSupp.Contact,
  ContSupp.AccountNo
from Contact1,
  ContSupp
where Contact1.AccountNo=ContSupp.Accountno
  and ContSupp.RecType='C'
  and Contact1.Company like 'Computer%'
order by Company
```

The resulting cursor was composed of some 29 records from my GoldMine database. Figure 7-8 depicts the results of said query against my GoldMine database. As always, at this point, we could select to **Output To ► Excel...**, or to build a Group in GoldMine Premium.

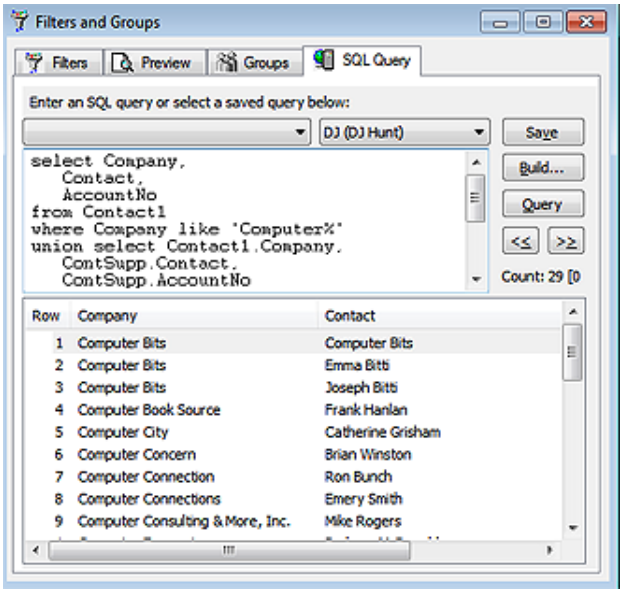


Figure 7-8

Okay, we all know that GoldMine Premium is the perfect application in that there are no bugs contained there in, however, for some reason many of my clients have come up with duplicate records in their **Contact2** table. We all know that this is a no, no. Each record in the **Contact1** table can have one, and only one, associated record in the **Contact2** table. To find out which contacts have duplicate **Contact2** table records, if any, I would run this query against their GoldMine database:

WARNING

*As the GoldMine Administrator, you really do not want any duplicate AccountNo records in your **Contact2** table.*

■ **Duplicates - Contact2 Records by AccountNo**

```
select count(*),
  AccountNo
from Contact2
group by AccountNo
  having Count(*) > 1
```

By the way, I just ran this against my **Contact2** table of 4,618 records today, and there were no duplicate records found. I must be getting better at caring for my database. Should your table contain duplicate records, you must immediately rectify the situation. There are a number of different ways of accomplishing the task all of which would require their own book to explain. If you find that you have duplicate records in the **Contact2** table, then you would want to get together with your GoldMine Partner to rectify the matter or you may hire our services if you currently do not have a GoldMine Partner.

Okay, they said that it couldn't be done within GoldMine, but I had always been able to prove them wrong. *"You cannot do an Update Query or a Delete Query from within GoldMine."*. Well, John Stillman caught wind of what I was doing, and is no longer permitting **Update** or **Delete** queries from within GoldMine in any form. He deemed that they are too dangerous, and I just have to agree with John on this one. Beginning with GoldMine Premium you can no longer do an **Update Query** or a **Delete Query** from within GoldMine.

Curse you John Stillman for taking away functionality in a program.

I could not imagine that you would not be interested in some of these other queries, so here are a few more for you to review:

- **Accounts with Bad E-mail Addresses**

```
select AccountNo,
  ContSupRef+Address1 as [E-Mail Address]
from ContSupp
where (Contact = 'E-mail Address')
  and isnull(ContSupRef, '') <> ''
  and (select
  case
    when ContSupRef is null
      or charindex('@.',ContSupRef + Address1) > 0
      or charindex('.@',ContSupRef + Address1) > 0
      or charindex('..',ContSupRef + Address1) > 0
      or charindex('"', ContSupRef + Address1) <> 0
      or charindex('(', ContSupRef + Address1) <> 0
      or charindex(')', ContSupRef + Address1) <> 0
      or charindex(',', ContSupRef + Address1) <> 0
      or charindex('<', ContSupRef + Address1) <> 0
      or charindex('>', ContSupRef + Address1) <> 0
      or charindex(';', ContSupRef + Address1) <> 0
      or charindex(':', ContSupRef + Address1) <> 0
      or charindex('[', ContSupRef + Address1) <> 0
      or charindex(']', ContSupRef + Address1) <> 0
      or right(rtrim(ContSupRef + Address1),1) = '.'
      or charindex(' ',ltrim(rtrim(ContSupRef + Address1))) > 0
      or len(ContSupRef + Address1)-1 <= charindex('.', ContSupRef + Address1)
      or ContSupRef + Address1 like '%@%@%'
      or ContSupRef + Address1 Not Like '%@%.%'
  then 0
  else 1
 end) = 0
```

- **Calendar - RecType A T O C, for UserID**

```
select *
from CAL
where RecType in ('A', 'T', 'C', 'O')
  and UserID = 'DJ'
order by OnDate
```

- **Casting - Date & Functions**

```
select top 100 cast(LastContOn  as varchar),
  datepart(month,LastContOn) [Month 1],
  month(LastContOn) [Month 2],
  LastContOn,
  *
from Contact2
where LastContOn is not null
```

- **Contact2 - Orphans**

```
select AccountNo
from Contact2
where AccountNo not in
  (select Accountno
  from Contact1)
```

- **Contacts - Not in any Group**

```
select AccountNo,
  Company,
  Contact
from Contact1
where AccountNo not in
  (select distinct Accountno
   from ContGrps)
```

- **Contacts - Record Age = 12 Months Old**

```
select Contact1.Company,
  Contact1.Contact,
  datediff(Month, Contact1.CreateOn, getdate()) as [Age in months],
  Contact1.CreateOn
from Contact1 with (NOLOCK)
left outer join Contact2 with (NOLOCK)
```

WARNING

*As the GoldMine Administrator, you really do not want any orphaned records in your **Contact2** table.*

Note

I have introduced a couple of differences here that you should be aware of. Notice the use of the specified join.

*Also, notice the use of the square brackets (**[A]**) instead of the single quotes (**'A'**) to delimit a string.*

```
  on Contact1.AccountNo = Contact2.AccountNo
where (datediff(Month, Contact1.CreateOn, getdate()) = 12)
```

- **Contacts - w/o Pending Activities**

```
select Contact
from Contact1
where AccountNo not in
  (select C1.AccountNo
  from Contact1 C1,
    Cal
  where C1.AccountNo=Cal.AccountNo)
```

- **Contacts - All include Primary E-mail Address if Exists**

```
select AccountNo,
  Company,
  Contact,
  (select top 1 ContSupRef+Address1
    from ContSupp as CS
    where CS.AccountNo = C1.AccountNo
     and Contact = 'E-mail Address'
     and Zip like '_1%'
     ) as Email
from Contact1 as C1
```

- **Count - Accounts by Representative**

```
select upper(Key4) as Representative,
count(distinct(Company)) as [# Contacts]
from Contact1
where Key1 like '%Client%'
  and Company > ''
group by Key4
```

- **Count - Activites (Calendar) by UserID within a Date Range**

```
select UserID,
  Count(AccountNo)
from Cal
where Ondate >= '9/1/2009'
  and OnDate <= '9/30/2009'
group by UserID
```

- **Count - Activites (History) by UserID within a Date Range**

```
select UserID,
  Count(AccountNo)
from ContHist
where Ondate >= '9/1/2009'
  and OnDate <= '9/30/2009'
group by UserID
```

- **Count - by State**

```
select State,
  count(State) as Count
from Contact1
group by State
order by State
```

- **Count - of Cities in State of Massachusetts**

```
select City,
  count(*) as [Count]
from Contact1
where State='MA'
  and City > ''
group by City
order by City
```

- **Count - Mail in Inbox for specific UserID**

```
select UserID,
  count(UserID)
from Mailbox
where UserID = 'DJ'
  and Folder = 'X-GM-INBOX'
group by UserID
```

■ **Duplicate - Contacts by Contact/Phone**

```
select count(*),
  Contact,
  Phone1
from Contact1
where Contact > ‘’
  and Phone1 > ‘’
group by Contact,
  Phone1
having count(*) > 1
order by Contact
```

■ **Duplicate - E-mail Addresses**

```
select AccountNo,
  Contact,
  ContSupRef,
  Address1
from ContSupp
where ContSupp.Contact = ‘E-mail Address’
  and ContSupRef in (select ContSupRef
    from Contsupp
    group by ContSupRef
    having count(ContSupRef)>1)
order by ContSupRef
```

■ **Mailbox - Usage**

```
select UserID,
  Folder,
  count(*)
from MailBox
where Folder not in (‘X-GM-INBOX’,
  ‘Sent’,
  ‘Filed’,
  ‘X-GM-DRAFTS’,
  ‘X-GM-FOLDERS’,
  ‘X-GM-GROUPS’,
  ‘X-GM-ICALINFO’,
  ‘X-GM-OUTBOX’,
  ‘X-GM-PROP-HTMLTAB’,
  ‘X-GM-HTMLTAB’,
  ‘X-GM-RULES’,
  ‘X-GM-TEMPLATES’,
  ‘X-GM-SUBSENT’,
  ‘X-GM-SUBFILED’,
  ‘X-GM-WEBIMPORT’,
  ‘X-GM-SMIME-CA’,
  ‘X-GM-TD-ITEMS’)
group by UserID,
  Folder
order by UserID,
  Folder
```

Note

*As of **GoldMine Premium 8.5.x** Notes are now stored as **Images**. Any query statement must account for this, and one must use the cast() and max() functions to query this information.*

■ **Notes - GoldMine Premium 8.5.x**

```
select OnDate as [Act Date],
  OnTime as [Act Time],
  Duration,
  Ref as [Reference],
  cast(cast(notes as varbinary(max))as varchar(max)) as Notes
from ContHist
where cast(cast(notes as varbinary(max))as varchar(max)) like ‘*** DJ%’
```

■ **Records Created within Last 30 days**

```
select *
from Contact1
where (CreateOn > { fn NOW() } - 30)
order by CreateOn
```

■ **Select - 2nd Group of 250 Records**

```
Select  Top 250 *
from Contact1
where RecID not in
  (select  Top 250 RecID
   from Contact1
```

```
  order by Contact )
order by Contact
```

- **Select - Convert Function - Specific Type History records on specific date**

```
select Contact1.Company,
  Contact1.Contact
from Contact1 inner join ContHist
  on ContHist.AccountNo  = Contact1.AccountNo
where ContHist.sRecType = 'T'
  and (ContHist.OnDate = convert(datetime, '2009-09-11 00:00:00', 102))
  and ContHist.ResultCode like 'C%'
```

- **Select - Date Function - Contact1 records created within the last 2 days**

```
select *
from Contact1
where (CreateOn > { fn NOW() } - 2)
```

- **Select - SQL Case Select - If Key1 empty use Key2, if both empty use Key1**

```
select
case Key1
when ''
then Key2
else Key1
end
from Contact1
```

- **Select - Substring Function**

```
select substring(ContSupp.ContSupRef, charindex('@', ContSupp.ContSupRef)+1, len(ContSupp.
ContSupRef)-charindex('@', ContSupp.ContSupRef))
from Contact1
join ContSupp on Contact1.AccountNo = ContSupp.AccountNo
where ContSupp.Contact = 'E-mail Address'
  and ContSupp.ContSupRef like '%DJ@DJHunt.US%'
```

- **Select - Substring Function**

  Prerequisite: ContSupp.Address1 = [MM/DD/YY]

```
select '20'+substring(Address1, 7, 2)+substring(Address1,1,2)+substring(Address1,4,2) as Date
from ContSupp
where '20'+substring(Address1, 7, 2)+substring(Address1,1,2)+substring(Address1,4,2) >=
'20090101'
  and '20'+substring(Address1, 7, 2)+substring(Address1,4,2)+substring(Address1,1,2) <=
'20091231'
```

- **Select - Sum/Case Functions - This one is particularly interesting for column display**

```
select UserID,
  sum(case when [FOLDER] = 'X-GM-INBOX' then 1 else 0 end) 'Inbox',
  sum(case when [FOLDER] = 'X-GM-OUTBOX' then 1 else 0 end) 'Outbox',
  sum(case when [FOLDER] = 'X-GM-TRASH' then 1 else 0 end) 'Trash'
from MailBox with (nolock)
group by UserID
order by UserID
```

- **Top 3 Companies by Sales**

```
select top 3
  C1.Company,
  C1.Contact,
  sum(cast(CH.Duration as Money)) as  TotalSales,
  count(CH.Duration) as SalesCount,
  CH.AccountNo
from Contact1 C1
  join Conthist CH
    on C1.AccountNo=CH.AccountNo
where CH.RecType = 'S'
  and CH.Duration like '%.%'
group by C1.Contact,
  C1.Company,
  CH.AccountNo
order by sum(cast(CH.Duration as Money)) desc
```

■ **Union Contact, Sales Pending & History**

```
select Contact1.AccountNo,
  Contact1.Company,
  Contact1.Contact,
  Contact1.Key1,
  Contact1.Address1,
  Contact1.Address2,
  Contact1.City,
  Contact1.State,
  Contact1.Zip,
  Cal.OnDate,
  Cal.Ref,
  cast (Cal.Number1 as varchar) as Value,
  cast (Cal.Duration as varchar) as Prob_Result
from Contact1,
  Cal
where Contact1.AccountNo = Cal.AccountNo
  and Contact1.Key1 = 'Client'
  and Cal.RecType = 'T'
union select Contact1.AccountNo,
  Contact1.Company,
  Contact1.Contact,
  Contact1.Key2,
  Contact1.Address1,
  Contact1.Address2,
  Contact1.City,
  Contact1.State,
  Contact1.Zip,
  ContHist.OnDate,
  ContHist.Ref,
  ContHist.Duration,
  ContHist.ResultCode
from Contact1,
  ContHist
where Contact1.AccountNo=ContHist.AccountNo
  and Contact1.Key1 = 'Client'
  and ContHist.sRecType = 'T'
order by Contact1.Contact,
  OnDate
```

Note

*It is critical that you have the same number & type of fields in your select statements on both sides of the **Union** or you will receive an error when you attempt your query.*

■ **Union Query - Information from Contact1 & ContSupp**

```
select Company,
  Contact,
  AccountNo
from Contact1
where Company like 'Computer%'
Union select Contact1.Company,
  ContSupp.Contact,
  ContSupp.AccountNo
from Contact1,
  ContSupp
where Contact1.AccountNo=ContSupp.Accountno
  and ContSupp.RecType='C'
  and Contact1.Company like 'Computer%' order by Company
```

That should give you enough queries to show you the endless possibilities. This, then, brings us to the conclusion of the **SQL Query** section of this chapter. However, you should understand by now that the **SQL Query** is a very, very powerful reporting tool. I suggest that you bone up on your **SQL Query** language, as it can only help you. I found a book from Sams' publishing, called *Teach Yourself SQL in 24 hours* by Ryan K. Stephend and Ronald R. Plew, to be extremely useful, and enlightening.

Groups

It is important to understand that a SQL Query is only the returning of data which exists in your database, and that it, in and of itself, cannot be activated or used as would a Filter or Group. Yet, the results of your SQL Query could be utilized to create a Group.

Let's say, as I recently had done, that you needed to know everyone in your GoldMine database that was still using GoldMine Standard Edition, and that had not, as yet, upgraded to GoldMine Corporate or Premium Edition. As I store this information as a **Detail** (**ContSupp** table), against my GoldMine database this would be the resulting query to pull the AccountNos for those records:

```
select distinct Contact1.AccountNo
from Contact1,
  ContSupp
where Contact1.AccountNo=ContSupp.AccountNo
  and Contact1.MergeCodes like '%E%'
```

```
and Contact1.MergeCodes not like '%N%'
and ContSupp.ContSupRef like '6.%'
and Contact1.AccountNo not in
  (select ContSupp.AccountNo
   from ContSupp
   where (ContSupp.ContSupRef like '7.%'
     or ContSupp.ContSupRef like '8.%'))
```

That query pulled some 171 records from my GoldMine database today, and, more importantly, that query can be saved using the **Save** button, Figure 7-9, and reutilized again and again.

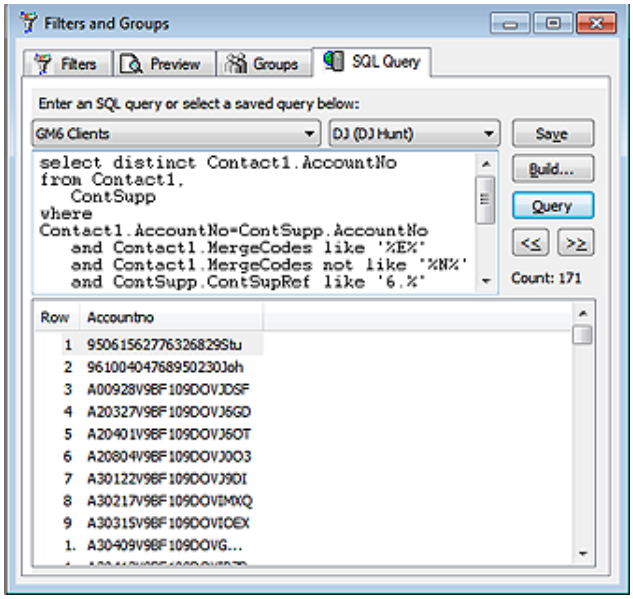


Figure 7-9

The reader will remember that a **Group** represents the data in GoldMine at a specific point in time. That point in time is the date, and the time that the group is generated. There are two grids on this dialog form under the **Groups** tab, Figure 7-10. The top grid, with the **Group Name**, **Code**, and the **Members** fields is where the user adds their Groups . As I mentioned earlier in this chapter, Groups are stored in the **ContGrps** table, and they are associated with the UserIDs. A right-click in this grid area will bring up a local menu from which the user may **Select User...**, from the GoldMine list of users, for which they may wish to employ or examine that UserIDs defined Groups. That, however, is only there for legacy as GoldMine Premium 8.5.x now has a **View Groups:** of UserID option from where one could just click on the drop arrow to select any of the defined UserIDs within GoldMine. More simply put, any user may use the groups of any other user if they have the **Access Rights** to do so.

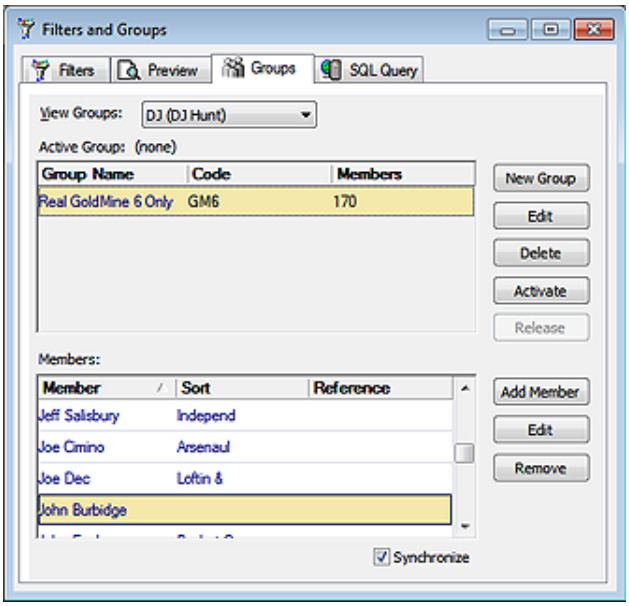


Figure 7-10

It is also from this local menu that the user may select to create a **New...** group or by simply clicking upon the **New Group** button. Doing so will bring up the **New Group** dialog form shown here in Figure 7-11. The creator is asked to supply a **Group Name:** in that field, however, the creator is not allocated much space to give the Group a really descriptive name. The creator is limited to only 24 characters with which to define this Group.

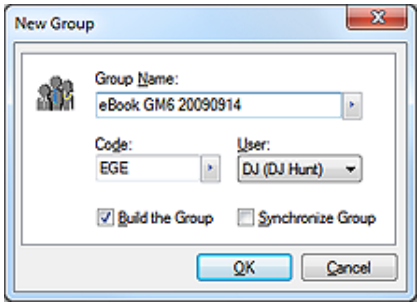


Figure 7-11

The **Code:** field, not to be confused with the **Activity Code** field when scheduling activities, is strictly a **Sort Order** field. GoldMine does not alphabetize the Group list by the **Group Name:** field, but instead, sorts the Group list alphabetically based on the **Code:** field. Additionally, as I stated earlier, not only may one view the Groups of another user, but they may also construct a Group in another users name. From the **User:** field

Note

*Key phrasing here: a SQL Query definition can be saved for later use, whereas, **Groups** must be rebuilt each and every time. There is no saving of the **Groups** definitions as there is for the **Filters** or the **SQL Query**.*

Tip

Keep in mind, as I am building a Group from the resulting cursor, that I only require the AccountNo. When building Groups only the AccountNo is required, and only one AccountNo will be added to the Group regardless of how many times it appears in the query results.

After all, do you really want to eBlast two of the same e-mails to the same Contact? I think not.

WARNING

*If this task is to be successful, you must go directly from the **SQL Query** tab to the **Groups** tab immediately following the clicking of the **Query** button as shown in Figures 7-9 & 7-10.*

Tip

*When naming anything anywhere in GoldMine, try to give a name that will describe what the action does. You want to make certain that, when you read the description a year from now, you, or anyone else, will be able to understand clearly what the Group consists of. As Groups are a static representation of data at a specific date/time, it would not be a bad idea to identify that date as a string date in your **Group Name:** description as I have done in Figure 7-11.*

drop-down list, select the name of the user for which the Group is being constructed. The default item in the drop list is the currently logged in GoldMine UserID, however, many organizations will simply build Groups under the **(public)** user.

The ☑ **Build the Group** checkbox is select in the default state. If this checkbox is selected, and the user were to click on the **OK** button, GoldMine would bring up the **Group Building Wizard** dialog form immediately, from which the user may construct the desired Group.

Figure 7-12

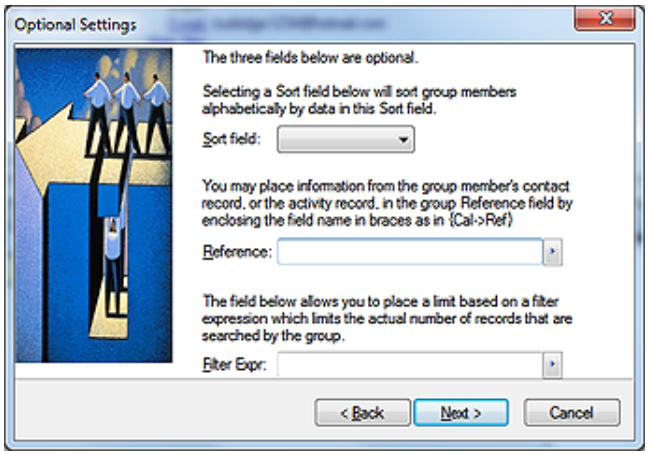


Figure 7-13

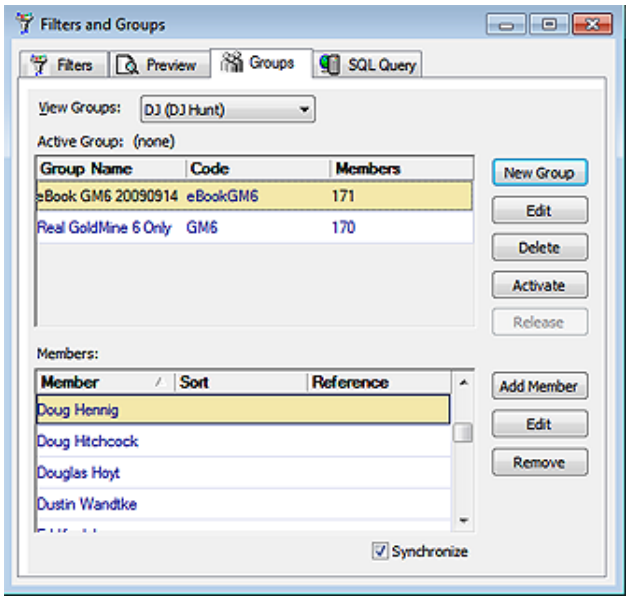


Figure 7-14

Groups are not normally synchronized to remote users via GoldMine in the default state. Should it be your wish, that your remote users have available the Groups which are created Server-side, then the creator of the group must select ☐ **Synchronize Group**. By default, this option is not selected, and one must be cognizant of this fact if they wish to have Groups synchronize to their Remote-side users.

Once the user has completed this dialog form, they must click on the **OK** button. I'll assume that you maintained the default setting, and will go directly into the **Group Building Wizard** as shown in Figure 7-12. Yes, I know, I have already preselected the ○ **SQL Query records** for this exercise, however, we all know that in this dialog forms default state, none of these options would have been selected. We are, after all, attempting to build a Group based on our previous SQL Query cursor, and these are the steps to accomplish that task.

Clicking on the **Next >** button will bring up the dialog form shown in Figure 7-13. I will discuss this dialog form in more detail coming up when I discuss something other that creating a Group as a result of your SQL Query cursor. Clicking on the **Next >** button will bring up the **Finish** dialog form, not shown.

Now, clicking on the **Finish** button would start the process, and GoldMine would build the Group from the SQL Query cursor. The **Filters and Groups** dialog form, at least against my GoldMine database, would look like this one shown here in Figure 7-14. It's just that simple, so even though you cannot save your Group building profile, you can save you SQL Queries, and rebuild your Group at any time in the future to capture the latest static information. Once built,

you could activate said Group, and utilize it for Reporting, eBlasting, Merge Documents, etc.

I would like to step back a bit now, and show you, what I probably should have shown you first, how to build your Groups by hand.

For this exercise, we will be building a Group that will match another of my, previously defined, select statements. I want all of the **Contact** records that have a **Primary E-mail Address**. To do this I must build the group based upon ○ **Supplemental contact data**, refer back to Figure 7-10 on the previous page. Before I do this, I would like to emphasize the various options by which one can build a group of contact records.

Groups could be built based on:

- ○ **Filtered records**
- ○ **Previewed records**
- ○ **SQL Query records**
- ○ **Tagged records**
- ○ **Search result**

- ○ **Scheduled calendar activities**
- ○ **Completed history activities**
- ○ **Supplemental contact data**

Once the user clicks on the **Next >** button, they will be stepped into the dialog form shown here in Figure 7-15.

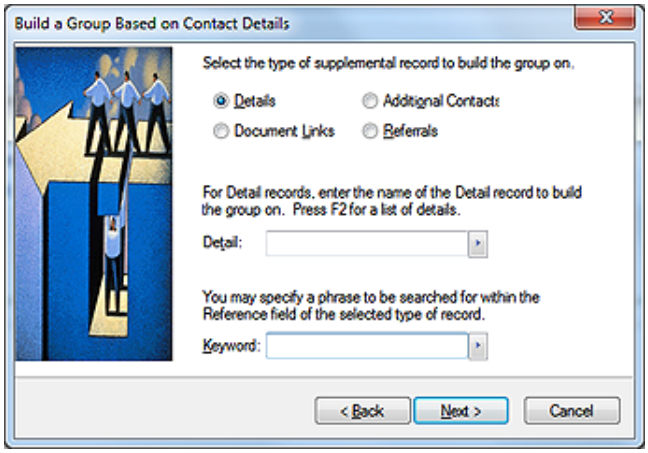


Figure 7-15

◉ **Details** will be selected by default, however, as mentioned in **The Tables** chapter, there are other record types maintained in the **ContSupp** table. The user, at this point, could as easily have chosen ○ **Document Links**, ○ **Additional Contacts**, or ○ **Referrals**, all of which represent supplemental contact data. I, however, want the default setting for this exercise.

The **Detail:** field F2 Lookup list is comprised of all of your defined **Details**. One could type into this field anything that they desire, however, unless it was an item from the F2 Lookup list, there would be little likelihood of finding any records that match the criterion specified. Hence, GoldMine will Force a match against the F2 Lookup List. For this exercise I chose E-mail Address from the F2 Lookup list.

For this exercise, there is nothing that I wish to enter into the **Keyword:** field. I, therefore, leave it blank, and simply click on the **Next >** button to bring us to the dialog form shown here in Figure 7-16. However, you should be aware that the **Keyword:** field search is a contains (**$**) statement so that you could have culled your results a little by including something like:

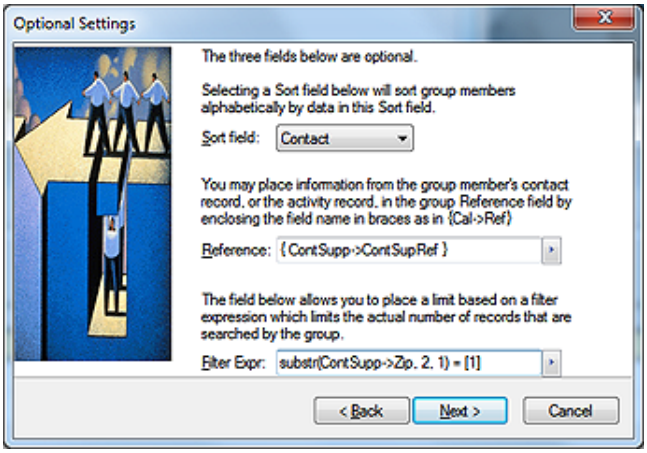


Figure 7-16

@DJHunt.US

The reader will notice that I have selected from the drop-down list of the **Sort field:** field, the **Contact** field on which to sort. The creator of the **Group** may select from any field contained within the **Contact1**/**Contact2** tables (refer to sidebar Note).

Note

No matter which field the creator of the Group selects to sort upon, only the first 8 characters of the selected field will be employed when creating the sorted group list.

The next field that one encounters is the **Reference:** field. If you will refer back to Figure 7-10, and look at the second grid, to the bottom of the figure, you will see columns for **Member**, **Sort**, and

Reference. Whatever expression the user enters here in the **Reference:** field, will be displayed in the **Reference** column of that grid. By default, GoldMine enters **{ContSupp->ContSupRef}**, which is the field that contains most of the actual e-mail address, therefore, I will accept the default in this case.

The last field on this page of the wizard, is the **Filter Expr:** field, and it is here that I must qualify those records that I wish in my list. Back in my **SQL Query** select statement, for this same Group, you'll remember that I discussed that the **Primary E-mail Address** is identified by having a **1** as the second byte of the **Zip** field in the **ContSupp** table (refer to **The Tables** chapter). As I am interested in only those **Contact** records that have a **Primary E-mail Address**, I must have a filter expression that will establish this same criterion. I have entered:

```
substr(ContSupp->Zip, 2, 1) = [1]
```

Once entered, I would click on the **Next >** button, and then on the **Finish** button. This produces the group result of 336 records when built against my GoldMine database.

Has anyone noticed a drawback in our discussion of the **Groups** tab, or the creation of a group? Not anywhere, in the wizard, did it ever ask me to save the profile. I've mentioned this earlier in this chapter. Each time I need to **Add Members**, or need to **Build a Group**, it must be done from scratch, stepping through the wizard each time. I must remember the fields that were previously selected, and the expressions that were used each time that I want to rebuild the **Group** to refresh it. In this exercise, it would have been wiser to have performed my **SQL Query** first, which could be a saved profile, and then to have built the **Group** based upon the resulting query cursor. It is much easier, and significantly faster. Hence, the reason that I covered **SQL Queries** before **Groups** in this chapter.

Record Tagging

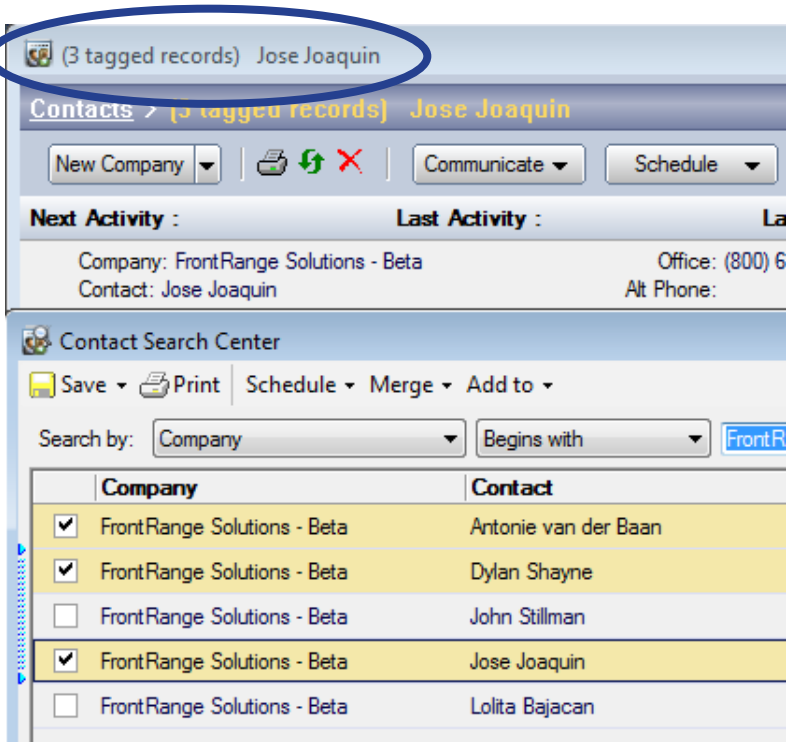


Figure 7-17

Groups may also be built based on ○ **Tagged records**, therefore, I thought that I would take a moment to explain how one goes about **Tagging** a record. As shown here, Figure 7-17, you should bring up the **Contact Search Center** dialog form. Holding the **Ctrl** key down or simply check the box preceeding the record to select (tag) a record. Notice that, in the **Contact** dialog form behind, the title has changed to include, **1 tagged records** when you select the first **Contact** record. You may tag as many records as you wish by finding the name in the **Contact Search Center**, holding the Ctrl key down, and selecting (left-clicking) the record with your mouse or by simply checking the box preceeding the record. You will notice that my titlebar shows that I have **3 tagged records**, and, as they are all on the same **Contact Search Center** screen, you can see the check marks next to those that I have tagged. You must leave the **Contact Listing** window open until after you have built your **Group** based upon ○ **Tagged records**.

You may also use these tagged records just as they are without building a **Group**. That's right, once **Tagged**, you may use this as your **Active Filter** for reports, merge documents, and e-mail merges or just anything that utilizes a **Filter/Group**. This will be your **Active Filter** until you close your **Contact Search Center**.

Relationship Tree

We've talked about the **Relationship Tree** elsewhere in this book on a number of occasions, but did you realize that a well structured Relationship Tree can also be used as an **Active Filter**? That's right. Take a look at this **FrontRange** tree in Figure 7-18. Notice that I am only showing you two of my many FrontRange Sub Folders, however, this should be enough to illustrate my point.

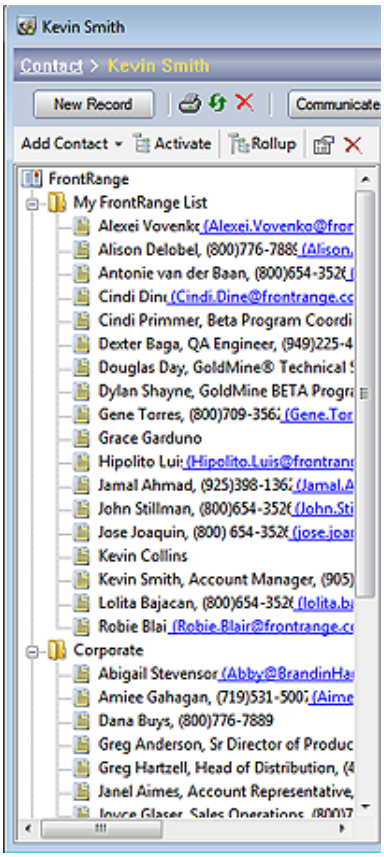


Figure 7-18

For argument sake, let's just say that I wanted to eBlast to everyone at FrontRange. I would simply highlight the **Book** level folder (**FrontRange**), And then right-click on that highlighted branch to bring up the local menu. From that local menu, I would then select **Activate Relationship Tree** or you could simply click on the **Activate** button on the toolbar running across the top of the relationship tree list.

Notice the titlebar in Figure 7-18 which currently reflects the Active Contact record belonging to **Kevin Smith**. After one has activated a tree, or branch of a tree, the titlebar would now indicate, per my example: **Section: FrontRange; Steve Salas**. Now go ahead and release that **Active Filter** by again selecting the local menu, but this time selecting **Release** from the local menu or the toolbar across the top of the relationship tree list.

Let's look at another data gathering feature of the Relationship Tree while we are here. Prior to the Relationship Tree (Organization Tree), people would enter all of the contacts belonging with a single organization under the **Additional Contacts** (**Others**) tab. This way all of the history for one organization, regardless of who it was with, would appear under the one **History** tab on that record. With the introduction of the Relationship Tree, then called Organization Tree, FrontRange asked us to begin the move to an Accountcentric solution by creating a single contact record for each organization member, and then relating them via the Relationship Tree.

Good enough, but then people wanted to know: What about the corporate history? It would now all be individual history. No problem on the SQL backend. Simply highlight, let's say, the **Corporate** branch of my tree. This will take you to the linked record for that branch. Now click on the **Rollup** button in the toolbar or **Roll up ALL Section's Contacts** from the local menu, your choice. GoldMine will now rollup all of the history for all contact records in the branch under the **History** tab of the **Active Contact** record. Sweet!

8

The Tables

In This Chapter

GoldMine is comprised of a number of related tables. These tables are, for the most part, referentially linked in some way, although there are quite a few independent tables as well.

FrontRange has assured me that the table structures for this, the second release of GoldMine Premium 8.5.x, has significantly changed the data structure, and that I would need to rewrite this chapter in my books. FrontRange now ships SQL Server 2008 for Workgroups with GoldMine Premium 8.5.x. After review the tables covered in this chapter, I saw very few changes to the schema, although those that are there are certainly significant.

On the next few pages I will give you the table structures, their default indexes, and any relationships as they exist as of this writing against GoldMine Premium 8.5.1.12. FrontRange is constantly updating GoldMine, and although they try not to make structural changes, it occasionally becomes necessary for the inclusion of some new functionality (aka Notes). As I give you these structures, if I have any insight about the particular table, I will also impart that information to you.

In this chapter I will examine what I feel are the most prominent tables used within the GoldMine Premium product. At a later date or edition, I may decide to add more tables to this chapter.

Contact1

Contact1 Table

Indexes

CN1RECID	RecID (Unique, Non-Clustered)
CONTACC	AccountNo, RecID (Non-Unique, Non-Clustered)
CONTCITY	u_City, AccountNo, RecID (Non-Unique, Non-Clustered)
CONTCNTY	u_Country, u_State, AccountNo, RecID (Non-Unique, Non-Clustered)
CONTCOMP	u_Company, u_Contact, RecID (Non-Unique, Non-Clustered)
CONTCURTAINING	Owner, Status, AccountNo (Non-Unique, Non-Clustered)
CONTKEY1	u_Key1, AccountNo, RecID (Non-Unique, Non-Clustered)
CONTKEY2	u_Key2, AccountNo, RecID (Non-Unique, Non-Clustered)
CONTKEY3	u_Key3, AccountNo, RecID (Non-Unique, Non-Clustered)
CONTKEY4	u_Key4, AccountNo, RecID (Non-Unique, Non-Clustered)
CONTKEY5	u_Key5, AccountNo, RecID (Non-Unique, Non-Clustered)
CONTLAST	u_LastName, u_Contact, RecID (Non-Unique, Non-Clustered)
CONTNAME	u_Contact, u_Company, RecID (Non-Unique, Non-Clustered)
CONTPHON	Phone1, u_Contact, AccountNo, RecID (Non-Unique, Non-Clustered)
CONTSTAT	u_State, u_City, AccountNo, RecID (Non-Unique, Non-Clustered)
CONTZIP	Zip, AccountNo, RecID (Non-Unique, Non-Clustered)

Relationships

Contact2.AccountNo	One-to-One	Contact1.AccountNo
ContHist.AccountNo	Many-to-One	Contact1.AccountNo
ContSupp.AccountNo	Many-to-One	Contact1.AccountNo
Cal.AccountNo	Many-to-One	Contact1.AccountNo
Mailbox.AccountNo	Many-to-One	Contact1.AccountNo
Cases.AccountNo	Many-to-One	Contact1.AccountNo
Notes.AccountNo	Many-to-One	Contact1.AccountNo

Structure

AccountNo	VarChar	20	Account Number - not Null*
Address1	VarChar	40	1st Address Line
Address2	VarChar	40	2nd Address Line
Address3	VarChar	40	3rd Address Line
City	VarChar	30	City
Company	VarChar	40	Company Name
Contact	VarChar	40	Contact Name
Country	VarChar	20	Country
CreateAt	VarChar	5	Creation Time
CreateBy	VarChar	8	Created by UserID
CreateOn	DateTime	8	Creation Date
Dear	VarChar	20	Dear/Salutation
Department	VarChar	35	Contacts Department
Ext1	VarChar	6	1st Phone Extension
Ext2	VarChar	6	2nd Phone Extension
Ext3	VarChar	6	Fax Extension
Ext4	VarChar	6	3rd Phone Extension
Fax	VarChar	25	Fax Number
Key1	VarChar	20	Key 1
Key2	VarChar	20	Key 2
Key3	VarChar	20	Key 3
Key4	VarChar	20	Key 4
Key5	VarChar	20	Key 5
LastDate	DateTime	8	Last Modified Date
LastName	VarChar	15	Contacts Last Name
LastTime	VarChar	5	Last Modified Time
LastUser	VarChar	8	Last Modified by UserID
MergeCodes	VarChar	20	Record Merge Codes
Notes	Text	16	Legacy Notes
Owner	VarChar	8	Record Owner
Phone1	VarChar	25	1st Phone Number - not Null
Phone2	VarChar	25	2nd Phone Number
Phone3	VarChar	25	3rd Phone Number
RecID	VarChar	15	Record ID - not Null
Secr	VarChar	20	Secretary
Source	VarChar	20	Lead Source
State	VarChar	20	State
Status	VarChar	3	GoldMine Status**
Title	VarChar	35	Contacts Title
U_City	VarChar	30	Upper Case City - not Null
U_Company	VarChar	40	Upper Case Company Name - not Null***
U_Contact	VarChar	40	Upper Case Contact Name - not Null
U_Country	VarChar	20	Upper Case Country - not Null
U_Key1	VarChar	20	Upper Case Key1 - not Null
U_Key2	VarChar	20	Upper Case Key2 - not Null
U_Key3	VarChar	20	Upper Case Key3 - not Null
U_Key4	VarChar	20	Upper Case Key4 - not Null
U_Key5	VarChar	20	Upper Case Key5 - not Null
U_LastName	VarChar	15	Upper Case Last Name - not Null
U_State	VarChar	20	Upper Case State - not Null
Zip	VarChar	10	Zip Code/Postal Code

*** The AccountNo field is comprised of**

Position 1 thru 6	Creation date in YYMMDD format (For Y2K compliance, positions 1 and 2 would store A9 for the year 2009)
Position 7 thru 11	Seconds since midnight
Position 12 thru 17	Randomly generated number
Position 18 thru 20	First three characters of Contact or Company name if Contact empty

**** The Status field is comprised of**

Position 1	U (US phone format) or I (International phone format)
Position 2	Curtain level (0 = None | 1 = Partial | 2 = Full | 3 = Semi-Partial)
Position 3	1 indicates a record alert is active for the contact record

*****GoldMine employs a mirror field for indexed fields**

These mirrored fields are included in the Indexes which helps to speed up searches against a SQL back-end, be it Microsoft or Firebird. These fields are auto-populated via a SQL Trigger with the upper case version of the contents in their parent field.

The **Contact1** table has one record for each primary contact in GoldMine. Both the **AccountNo** and the **RecID** are unique identifiers in the table with the **AccountNo** being a **Primary Key**, and the **RecID** being a **Candidate** Key. The **AccountNo** is used as the **Primary Key** in all of the relationships in other tables, while the **RecID** uniquely identifies each record within any table.

The **Contact1** table maintains 16 separate indices at all times in its default state. Each index is employed when the user is performing a lookup for a particular piece of information making that lookup, even against a large database, extremely fast. FrontRange has been asked, and often, to include more indices (see sidebar Note). They have determined that these 16 are an optimal number of indexes. Any more, and performance could take a hit. **CN1RecID** was the last index added.

There are seven primary relationships which have been identified for the **Contact1** table. There is the one-to-one relationship that exists between the **Contact1** table and the **Contact2** table. There should never be more than one record in the **Contact2** table for each record in the **Contact1** table. If the GoldMine administrator notices that there are multiple records in the **Contact2** table having the same **AccountNo**, then it behooves them to rectify the matter. This must be accomplished using 3rd party software to maintain synchronization awareness. Remember, in an earlier chapter, I gave you this SQL select statement to identify any **Duplicate Contact2** records:

```
select count(*), AccountNo
from Contact2
group by AccountNo having Count(*) > 1
```

Additionally, there exists a one-to-many relationship between the **Contact1** table, the **ContHist**, **ContSupp**, **Cal**, **Mailbox**, **Cases**, and the **Notes** tables. This means that for each **AccountNo** in the **Contact1** table there can be multiple records in these other tables possessing the same **AccountNo** which relates the records in both tables. You can have many historical activities against one contact record, as you could have many scheduled activities against one contact record. The **ContSupp** is a unique table that serves multiple purposes. I'll discuss that momentarily.

One thing that I would like to discuss here is the **Contact1.MergeCodes** field. The information contained in this field is employed when merging to document templates, and should not be confused with the merge codes that I will discuss later when I talk about the **ContSupp** table, and it's E-mail Address records.

Note

*FrontRange has informed me that, unlike previous versions of GoldMine, users will be able to add their own indices through a **Firebird Database Manager**, or **SQL Server Management Studio**, and that GoldMine will pick up, and honor/use the new indices.*

I would advise that you try not to add any user defined index, unless it is absolutely required, as any increase in the index count is bound to have an adverse affect on performance. You may want to add the index one at a time, and have your staff work with them for a while to assure that their GoldMine performance is still at an acceptable level. If acceptable, then add the next index that you need, and test the performance again for another period of time, and so on.

Contact2

Contact2 Table

Indexes

CONTACT2	AccountNo (Non-Unique, Non-Clustered)
CN2RECID	RecID (Unique, Non-Clustered)

Relationships

Contact2.AccountNo	One-to-One	Contact1.AccountNo

Structure

AccountNo	VarChar	20	Account number - not Null
ActionOn	DateTime	8	Next Action on Date
CallBackAt	VarChar	8	Call back Time (unused)
CallBackOn	DateTime	8	Call back Date
CallBkFreq	SmallInt	2	Call back frequency
CloseDate	DateTime	8	Forecast Sale expected Close
Comments	VarChar	65	Comments
LastAtmpAt	VarChar	8	Last Attempt at Time
LastAtmpOn	DateTime	8	Last Attempt on Date
LastContAt	VarChar	8	Last Contacted at Time
LastContOn	DateTime	8	Last Contacted on Date
MeetDateOn	DateTime	8	Meeting on Date
MeetTimeAt	VarChar	8	Meeting at Time
NextAction	VarChar	65	Next Action
PrevResult	VarChar	65	Previous Results
RecID	VarChar	15	Record ID - not Null
UserDef01	VarChar	10	User defined field 1*
UserDef02	VarChar	12	User defined field 2*
UserDef03	VarChar	15	User defined field 3*
UserDef04	VarChar	12	User defined field 4*
UserDef05	VarChar	10	User defined field 5*
UserDef06	VarChar	10	User defined field 6*
UserDef07	VarChar	3	User defined field 7*
UserDef08	VarChar	10	User defined field 8*
UserDef09	VarChar	10	User defined field 9*
UserDef10	VarChar	10	User defined field 10*
UserDef11	VarChar	10	User defined field 11*
UserDef12	VarChar	10	User defined field 12*
UserDef13	VarChar	25	User defined field 13*
UserDef14	VarChar	10	User defined field 14*
UserDef15	VarChar	25	User defined field 15*
UserDef16	VarChar	10	User defined field 16*

*GoldMine User Defined Fields

GoldMine **UserDef01** thru **UserDef10** can never be deleted, whereas, **UserDef11** thru **UserDef16** you are permitted to delete. In all User Defined field cases you may not change the VarChar type, however, you are permitted to change the data size.

WARNING

*The number of records in the **Contact2** table should never exceed the number of records in the **Contact1** table. You are advised to check the state of these two tables regularly using the queries:*

```
select count(*)
from Contact1

select count(*)
from Contact2
```

The **Contact2** table has only two indexes. The first, **Contact2**, is used to maintain the relationship between the **Contact1**, and the **Contact2** tables, however, because of its Non-Unique status it is possible to have two **AccountNos** exactly the same in the **Contact2** table (see WARNING in sidebar).

This table should contain one record for each record in the **Contact1** table, however, the reverse does not hold true. You could have fewer **Contact2** records than you have **Contact1** records. When one created a new **Contact1** record, a new **Contact2** record was not created automatically in past versions of GoldMine. The **Contact2** record was only created when there was a need to store information in fields contained in the **Contact2** table. This should not occur in new GoldMine Premium installations, but is possible if you upgraded to GoldMine Premium from an older build of GoldMine. Occasionally it is possible, although unlikely, to have a **Contact2** record without having a matching **AccountNo** in the **Contact1** table. These are called **Orphans**. When this occurs the GoldMine Administrator is advised to use a 3rd party software application to find, and remove these orphaned records. I did supply you with a SQL Query to identify these orphans, but here it is just in case:

■ **Contact2 - Orphans**

```
select AccountNo
from Contact2
where AccountNo not in
  (select Accountno
  from Contact1)
```

As you add user defined fields to your GoldMine, these new user defined fields will be contained in the **Contact2** table, while their structural definitions will be maintained in the **ContUDef** (see next page) table. The **ContUDef** table will reside in the same database as the **Contact2** table.

ContUDef

ContUDef Table

Indexes

CONTUDEF	DBFName, Field_Name (Non-Unique, Non-Clustered)
CNURECID	RecID (Unique, Non-Clustered)

Structure

DBFName	VarChar	8	**CONTACT1** or **CONTACT2** - not Null
Field_Dec	SmallInt	2	Number of decimal places contained in the field
Field_Len	SmallInt	2	Field Length
Field_Name	VarChar	10	Field Name - not Null
Field_Pict	VarChar	20	Field Picture
Field_Type	VarChar	1	Field Type*
FieldDesc	VarChar	25	Field Description (Global Label)
FieldNo	VarChar	3	Field Number (Tab Order User Defined Fields)
FldOpts	VarChar	8	Field Options**
LocalLabel	VarChar	25	Local Label for the field
RAccess	VarChar	8	Read Access User or User Group
RecID	VarChar	15	Record ID - not Null
Status	VarChar	3	Field Status (Currently Unemployed)
WAccess	VarChar	8	Write Access User or User Group

* Field Type

C - VarChar
N - Numeric
D - DateTime
M - Text (Legacy Contact1.Notes)

** FldOpts

0 - Do not log changes
1 - Log changes

There is one record in the **ContUDef** table for each field in the **Contact1**/**Contact2** tables in any given contact database. The primary reason for including the **Contact1** fields in the **ContUDef** table is not to maintain the field definitions, as these are maintained in the data dictionary, but to allow for **Local Labeling** of the fields as well as the capability of maintaining the **Read**/**Write Access** properties which are also not maintained in the data dictionary.

ContHist

ContHist Table

Indexes

CONTHIST	AccountNo, OnDate, RecID (Non-Unique, Non-Clustered)
CONTHUSR	UserID, sRecType, OnDate, RecID (Non-Unique, Non-Clustered)
CNHRLINK	LopRecID, OnDate (Non-Unique, Non-Clustered)
CNHLSDT	LastDate (Non-Unique, Non-Clustered)
CNHRECID	RecID (Unique, Non-Clustered)

Relationships

ContHist.AccountNo	Many-to-One	**Contact1.AccountNo**
ContHist.LOpRecID	Many-to-One	**OpMgr.RecID**
ContHist.LOpRecID	Many-to-One	**Cases.RecID**
ContHist.RecID	One-to-One	**MailBox.LinkRecID**
ContHist.LinkRecID	One-to-One	**MailBox.RecID**

Structure

AccountNo	VarChar	20	Linked Contact AccountNo - not Null
ActvCode	VarChar	3	Activity Code
CreateAt	VarChar	5	Created at Time
CreateBy	VarChar	8	Creation UserID
CreateOn	DateTime	8	Created on Date
Duration	VarChar	14	Duration***
Ext	VarChar	5	Notes characteristics html or plain
LastDate	DateTime	8	Last Modified on Date
LastTime	VarChar	5	Last Modified at Time
LastUser	VarChar	8	Last Modified by UserID
LinkRecID	VarChar	15	Linked MailBox Record ID
LOpRecID	VarChar	15	Linked Opportunity Manager Record ID/ Case Manager Record ID - not Null
Notes	Image	16	Notes
OnDate	DateTime	8	Activity on Date
OnTime	VarChar	5	Activity on Time
RecID	VarChar	15	Record ID - not Null
RecType	VarChar	10	Record Type*
Ref	VarChar	80	Reference****
ResultCode	VarChar	3	Result Code
SRecType	VarChar	1	1st character of RecType - not Null
Status	VarChar	2	GoldMine field**
Units	VarChar	20	Units of a Forecasted Sale
UserID	VarChar	8	User - not Null

* The following are possible values for the RecType field:

A	Appointment	**O**	Other	**CO**	Outgoing call
C	Phone call	**S**	Sale	**MG**	E-mail message
D	To-do	**T**	Next Action	**MI**	Received e-mail
E	Event	**U**	Unknown	**MO**	Sent e-mail
F	Literature fulfillment	**CC**	Call back	**RO**	Audit Override
L	Form	**CI**	Incoming call	**RS**	Audit New
M	Sent message	**CM**	Returned message		

** Status

1st character **Flag**
2nd character **1** if there are notes in the **Notes** field.

*** Duration

This takes the form of **hh:mm:ss** for most activities or it may be the string form of a number such as **[1.5]**.

**** Ref

The contact linked to the activity is stored at the end of the Ref field preceded by (oc:.
i.e. Sample for eBook (oc: DJ Hunt)

An additional note that should be conveyed, pertains to the **LOpRecID** field. Programmers should be aware that this field must contain a value an is set to **Not Null**. If there is no actual link to the **OpMgr**/**Cases** table, then the programmer is required to stuff this field with a space, plus the first **14** characters of a newly generated **RecID**.

Also, for the programmers among us, the **OnTime**, **CreateAt**, and **LastTime** fields all contain values based upon the 24 hour clock. Endusers may use the **FmtTime()** function (Appendix A) to display this value based on the 12 hour clock.

There may be many history records for any given **AccountNo** number, as well, there may be history activities that have no **AccountNo** value. History activities without an **AccountNo** value in the field are unlinked activities and may be considered orphans.

The **Duration** field may be **.null./empty**, may contain a character based numeric value (**1.25**), or may contain a time designation (**01:30:47**). All of these are acceptable values in this field. One could expect to see **357.25** in the Duration field if the **sRecType** field contained an **S**. On the other hand, if the **sRecType** field contained an **A** value, then the value in the **Duration** field could be, either, **1.25**, or **01:15:00**. as both are acceptable.

Note

*Notice that the **Notes** field takes on new characteristics in GoldMine Premium as of 8.5.1.12.*

Note

*The **3rd byte** of the **RecType** field is used to store the **Success** (**space**) or **Unsuccessful** (**U**) for the activity.*

*The **4th byte** of the **RecType** field is used to store the privacy (**P**) value if the activity is marked as **Private**. The 4th byte may also be used when the Completed activity was the result of an Auto Generate **RSVP** (**R**) selection on the scheduled activity.*

*The **5th byte** of the **RecType** field is used to store any color coding for the activity as carried over when completing a Scheduled Activity.*

ContSupp

ContSupp Table

Indexes

CONTSUPP	AccountNo, RecType, u_Contact, RecID (Non-Unique, Non-Clustered)
CONTSPFD	RecType, u_Contact, u_ContSupRef (Non-Unique, Non-Clustered)
CNTSUPADDR	RecType, u_Address1 (Non-Unique, Non-Clustered)
CNSRECID	RecID (Unique, Non-Clustered)
CSUPREFIDX	u_ContSupRef (Non-Unique, Non-Clustered)

Relationships

ContSupp.AccountNo	Many-to-One	**Contact1.AccountNo**
ContSupp.AccountNo	Many-to-Many	**ContSupp.LinkAcct**

Structure

AccountNo	VarChar	20	Linked Contact AccountNo - not Null
Address1	VarChar	40	Additional Contact Address 1
Address2	VarChar	40	Additional Contact Address 2
Address3	VarChar	40	Additional Contact Address 3
City	VarChar	30	Additional Contact City
Contact	VarChar	40	Contact Name/Detail
ContSupRef	VarChar	35	Reference
Country	VarChar	20	Additional Contact Country
Dear	VarChar	20	Dear/Salutation
Ext	VarChar	6	Phone Extension
Fax	VarChar	20	Fax Number
LastDate	DateTime	8	Last Modified on Date
LastTime	VarChar	5	Last Modified at Time
LastUser	VarChar	8	Last Modified by UserID
LinkAcct	VarChar	20	Linked AccountNo
LinkedDoc	Text	16	Linked Document
MergeCodes	VarChar	20	Additional Contact or E-mail Merge Codes
Notes	Text	16	Notes
Phone	VarChar	20	Phone Number
RecID	VarChar	15	Record ID - not Null
RecType	VarChar	1	Record Type* - not Null
State	VarChar	20	Additional Contact State
Status	VarChar	4	GoldMine field**
Title	VarChar	35	Contact Title/Referrals account number
U_Address1	VarChar	40	Upper Case Address1*** - not Null
U_Contact	VarChar	40	Upper Case Contact Name*** - not Null
U_ContSupRef	VarChar	35	Upper Case Reference*** - not Null
Zip	VarChar	10	Additional Contact Zip/Postal Code

* The following are possible values for the RecType field:

A	Record Alerts	**L**	Linked document
C	Additional contact record	**O**	Relationship Tree
E	Automated process attached event	**P**	Profile record/extended profile record
H	Extended profile header	**R**	Referral record
V	CS_Version		

** Status

1st byte **HTML On/Off**
2nd byte **1** notes in the **Notes** field
3rd byte **Wrap On/Off**
4th byte **MIME On/Off**

***GoldMine employs a mirror field for indexed fields

These mirrored fields are included in the Indexes which helps to speed up searches against a SQL back-end, be it Microsoft or Firebird. These fields are auto-populated via a SQL Trigger with the upper case version of the contents in their parent field.

Now we have the motherload of tables. Not only can there be a 1 to many relationship between the **Contact1** table and this, the **ContSupp** table, but it can be a 1 to many relationship many times based on the **ContSupp.RecType** value. You see, the **ContSupp** table is used for many different record types which each have their own display area within GoldMine. Over the following pages I will discuss the various **ContSupp** record settings based on the various **ContSupp.RecType** values as shown above. You will see that the same field may be used differently based on the **ContSupp.RecType** value.

Record Alert

Each field in the **ContSupp** table is capable of holding a variety of information, and again, it is all dependant upon the **ContSupp.RecType** field value. The first **ContSupp.RecType** that I will examine is that of **A**. When a **Record Alert** is attached to a **Contact** record, the actual alert for that contact is stored in the **ContSupp** table, while the characteristics of the alerts are stored in the **InfoMine** table (not to be covered in this book), and better known to most as the **Knowledge Base**. In the **ContSupp** table itself, the record pointer to the alert stores very little information. These represent the fields that were populated for a typical alert:

AccountNo - stores the relational link back to the appropriate record in the **Contact1** table.

RecType - will always be **A** for this type of record.

Notes - contains the **Alert Code**, the **UserID** and any triggers. In example:

```
~TST~DJ+chr(9)+1+chr(10)
```

LastUser, **LastDate**, **LastTime**, and **RecID** - are all employed in the normal manner.

Add'l Contacts

These **Additional Contacts** should not, in any way, be confused with the main or **Primary Contacts**, but, instead, are related to the **Primary Contact** as **Additional Contacts**. These records contain a **ContSupp.RecType** of **C**. For this record type all of the fields are, pretty much, used as the field labels would indicate. The few exceptions are:

Recommendation

*This author does not recommend that any organization utilize the **Additional Contacts** option within GoldMine. I always ask my clients to create separate records for each **Contact**, and **Contact Location** (i.e. **Home** or **Office**). If there is any relationship between these records I ask that one use the **Relationship Tree**, in GoldMine Premium, to graphically represent these relationships.*

AccountNo - stores the relational link back to the appropriate record in the **Contact1** table.

RecType - will always be **C** for this type of record.

ContSupRef - will hold any information supplied through the **Ref** field in the dialog form when **Adding/Editing** an **Additional Contact** record. GoldMine also utilizes this field when it is creating an **Additional Contact** from a deletion of a merged **Contact** record. In this case, GoldMine would populate this field with the value **Duplicate Record**.

Status - 1st character = **1** if there is an associated/related **E-mail Address** which would also be contained within this table as another record with a **ContSupp.RecType** of **P**. Once populated, the **ContSupp.Status** field will remain populated, and the bytes will be turned on (**1**) or off (**0**) as necessary. However, the **ContSupp.Status** field will remain **.null.** until such time as either an **E-mail Address** is added, or a note is added in the **ContSupp.Notes** field. The 2nd character = **1** if there is an associated note or it could have a **0**, once the field has been populated, if there are no notes associated with the particular record.

MergeCodes - This field is used for merge codes that are to be applied against Additional Contacts when merging to word document templates.

Note

*Please remember that the **Contact1.MergeCodes** field is for merge codes that pertain to **Word Document Template** merges, and should not be confused with the **ContSupp.MergeCodes** field that is associated with, and used for the **E-mail Template** merges (discussed later in this chapter).*

LastUser, **LastDate**, **LastTime**, and **RecID** - are all employed in the normal manner.

Automated Process Tracks

Continuing on, and in alphabetical order, the **RecType** of **E** would be next. Whenever an **Automated Process Track** is attached to a **Contact** record, a **ContSupp** record of this type is created to maintain the current state of any attached Track. There are a few fields in the record that maintain information.

AccountNo - stores the relational link back to the appropriate record in the Contact1 table.

RecType - will always be **E** for this type of record.

Contact - maintains the Automated Process attachment date & time information, and takes the form of:

```
YYYYMMDDHH:MM:SS ( 2009091509:35:19 )
```

Title - stores the **Tracks.TrackNo** field information of the related track.

ContSupRef - will maintain the current position of the track within the process, or, probably better understood as the next **Sequential Event** that is to be tested. The information for this value is drawn from the **Tracks.Name** field.

Phone - this field maintains the **Tracks.Options** flags for the specified track. The information is a set of 5 digits, in a character based field, representing the state of the flags. In example:

```
10001
```

Ext - maintains the **Track.EventNo** for the **Next Event**.

LinkAcct - maintains the **Last Event** date in the form of:

```
YYYYMMDD ( 20090915 )
```

Address1 - is used to maintain the **Tracks.Name** from the parent track.

City - maintains the **Next Event** (**Tracks.Name**), in English, for the **Next Event**.

State - maintains the **Track.EventNo** for the **Next Event**.

Zip - maintains the **UserID** of the individual who attached this **Track** to the **Contact** record.

Country - is used to maintain the **Trigger** basis (**Tracks.TriggerTyp**), in English, for the **Next Event**.

LastUser, **LastDate**, **LastTime**, and **RecID** - are all employed in the normal manner.

Headers

Next we should look at the **RecType** of **H**. This record type is where the **Setup** information is maintained for the various **Details** for which a user creates, and for which extended information has been established. GoldMine looks at this record to ascertain which fields, of the extended fields, to display, and what is to be displayed as their label.

AccountNo - stores the relational link that is used by **ContSupp P RecType** to relate the setup criterion for the **Detail**. This might take the form of:

```
HxD5UZ1F8+#
```

RecType - will always be **H** for this type of record.

ContSupRef - will contain the name of the **Detail**, up to 20 characters along with the final 15 characters being allocated to the tab name if one has been established for this particular **Detail**. In example:

```
Partner          Part&ner
Handango           MyDiabetes
Credit Card      Credit Card
GoldMine         GoldMine
```

Note

*The double asterisk (**) Instructs GoldMine to not display the field on the dialog form when displayed for this **Detail** type. Anything else following the equal sign (=) Indicates that the field is to be displayed and that this information is to be displayed as the label for the equivilent field.*

*I would add that **Details Plus** exposes more **ContSupp** fields to be used for **Details**, as well as being capable of allowing you to design the layout of those fields on the screens. Additional information can be found at:*

http://www.Solica.com

Phone - this field appears to always carry a **0**, and I have yet to determine its significance.

Notes - is where the extend field label association is maintained, and contains the true field name along with its local label for all eight of the extended detail fields utilitized by GoldMine Premium. In example from the GoldMine **Detail** shown above:

```
Title   =HDA#
LinkAcct=PoR
Country =**
Dear    =**
Fax     =**
Zip     =**
Ext     =**
State   =Expiration
MergeCodes=**
Address1=Serial #
Address2=:
Address3=**
SaveCols=1
```

All information preceeding the equal sign (**=**) consists of exactly 8 characters except for the **MergeCodes** label which is, obviously, 10 characters. Prior to GoldMine Premium 8.5.1.12 only the **Title**, **LinkAcct**, **Country**, **Zip**, **Ext**, **State**, **Address1** & **Address2** fields could be utilized in the **Extended Details** for a total of eight extended fields. Today we have the capability of 12 extended fields within a GoldMine **Detail** with the last entry telling GoldMine whether or not to save the user defined column settings for that **Detail** type.

LastUser, **LastDate**, **LastTime**, and **RecID** - are all employed in the normal manner.

Linked Documents

I can now look at the **RecType** of **L**, and this record type is utilized for the **Linked Documents**.

AccountNo - stores the relational link back to the appropriate record in the **Contact1** table.

RecType - will always be **L** for this type of record.

Contact - this field contains 10 spaces, the document extension type, 8 spaces, and the link date. In example:

```
Expression: padr(UserID, [ ], 10) + document extension + space(8) + dtos(date())
     Value: [DJ       .dot       20090930]
```

Title - holds the document type. Based on the information in the **Contact** field example above, this document type might be **Microsoft Office Word 97 - 2003** as this is the association that my computer holds for a .dot extension.

ContSupRef - contains the descriptive **Document Name:** entered when the link was created or it might be the **Subject:** of the E-mail from which the linked document was derived.

Dear - this field maintains the link document creation date & time information, and takes the form of:

Expression: **YYYMMDDHH:MM:SS**
Value: **2007031610:43:03**

LinkAcct - is used when the linked document is the result of a link to an e-mail received, and contains the **MailBox.RecID** of the linked e-mail message, otherwise, this field will probably remain **.Null.**.

Address1/**Address2**/**Address3** - these fields carry the physical document name, and the path to that document. Remembering that each field is 40 characters, **Address1** takes characters **1** through **40** of the path while **Address2** takes characters **41** through **80** of the path, with any remaining characters being dumped into the **Address3** field.

City - this field is broken up into 3 concatenated sections. The first section is the eight character UserID. If the UserID is DJ, 2 characters, then this first section will be DJ plus six spaces. The next section contains the string representation of the date that this record was created. This might look like 20070327. The next section, 7 characters long, is the creation time of this record. It is based on a 12 hour clock, and it includes a space if it is not a 2 digit hour. Here are a couple of possible representations of a value from this field:

DJ 20090927 9:31am
DJ 2009092710:49am

Status - this is, again, a flag field. This first byte may indicate that this is an attachment (**A**). The second byte is set to **1** to indicate that there are notes associated with this record or set to **0** when there are no notes associated with this record.

LinkedDoc - this field contains most of the sync information about the file, plus the actual file that is launched when instructed to do so by a user. The information in this field could look like:

~~SYNC=1
~~CREATETIME=200803120721
~~FILENAME=Y:\goldmine\mail box\dhunt\DJ\2008\03\image001.jpg
~~SYNCSTAMP=200803120721

LastUser, **LastDate**, **LastTime**, and **RecID** - are all employed in the normal manner.

Note

*Please note that each of the lines in the **ContSupp.LinkedDoc** field is followed by **chr(10)**, and not the normal **chr(13)+chr(10)**. Additionally, and most importantly for you programmers out there, the last line must end with a **chr(10)**.*

Relationship Tree

Note

*Except in rare circumstances, I always remove the link between the **Books** (Top Level), and **Folders** (Sections) to any given contact record. Usually, I only link the **Contact** (Page) to a **Contact** record. In the sample **Relationship Tree** that ships with GoldMine Premium, I noticed that FrontRange follows this paradigm as well.*

*Based on the **Relationship Tree** shown to the right in Figure 8-1, I think that you can easily see why I do not link the **Books** and **Folders**.*

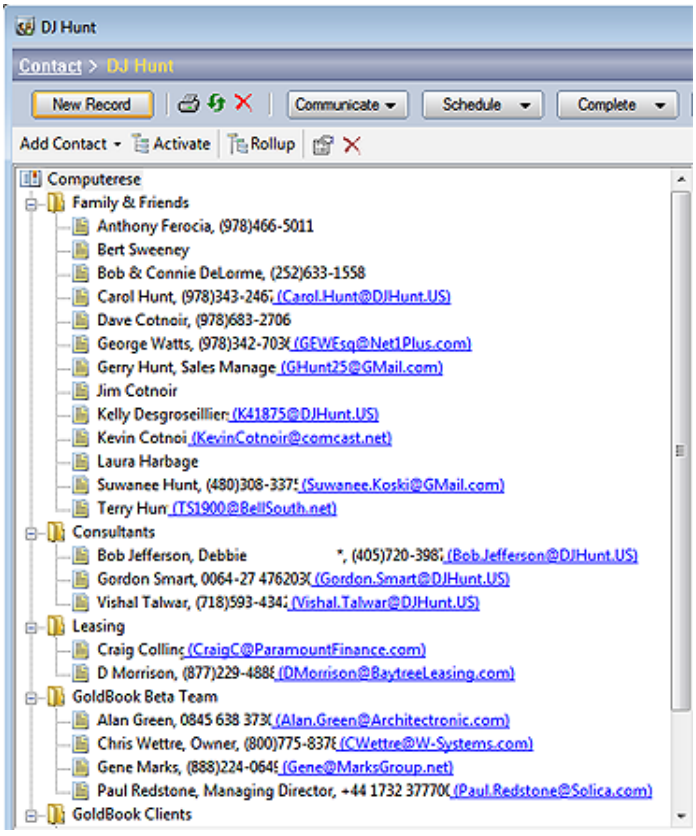


Figure 8-1

The next **RecType** that I will examine with you is that of **O** for the **Relationships Tree** record. This represents yet another totally different usage of each of the fields in the **ContSupp** table. Before I examine the fields for this type of **RecType**, and what they contain, I would refer you to Figure 8-1. Figure 8-1 is a **Relationship Tree** that I created against the my GoldMine database. Using this **Relationship Tree** as the basis for the information contained in the various **ContSupp** fields, let's examine it in detail.

AccountNo - if this field is populated, it represents a link between the **Relationship Tree** entity, and a particular contact record.

RecType - these will always have a type of O.

Contact - will contain a uniquely generated **AccountNo** for the **Book** (top level), and all entities contained within this book will carry this value in the **ContSupp.Contact** field.

ContSupRef - contains the sort order for the **Relationship Tree**. The data in this field will be a numeric value representing a **Book** (top level), or **Folder** (section), or a numeric value, and as much of the contact name as the **35** character limit of this field will permit. This will vary with the number of SubSections that may be involved.

For the **Relationship Tree** shown in Figure 8-1 on the previous page, these are the values contained within the **ContSupRef** field sorted as they would be for the **Relationship Tree** ordering:

10000
1000001000
1000001000 Anthony Ferocia
1000001000 Bert Sweeney
1000001000 Bob & Connie DeLorme
1000001000 Carol Hunt
1000001000 Dave Cotnoir
1000001000 George Watts
1000001000 Gerry Hunt
1000001000 Jim Cotnoir
1000001000 Kelly Desgroseilliers
1000001000 Kevin Cotnoir
1000001000 Laura Harbage
1000001000 Suwanee Hunt
1000001000 Terry Hunt
1000001500
1000001500 Bob Jefferson
1000001500 Gordon Smart
1000001500 Vishal Talwar
1000002000
1000002000 Craig Colling
1000002000 D Morrison
1000002500
1000002500 Alan Green
1000002500 Chris Wettre
1000002500 Gene Marks
1000002500 Paul Redstone
1000003000
1000003000 Damian Schwarz
1000003000 Debbie Green
1000003000 Eric Soloff
1000003000 Gary Zimmerman
1000003000 Rick Kabra

Note

*It is because of the **35** character limitation of the **ContSupRef** field that FrontRange imposes a **5 SubSection** limitation for the **Relationship Tree**. Based on the schema that they have developed this is the maximum that the schema will permit.*

Ext - this field holds the icon type to be displayed in the **Relationship Tree**. If this is the record for the **Book** level, then this field will contain a **1**. If this record is for a **Section** level, then this field will hold a **3**. Whereas, if this record is for a **Contact** level it will contain a **6**.

Note

*Please note that additional information displayed is the **Relationship Tree** is not stored nor maintained in the record, and is, however, generated by GoldMine for the graphical display of the tree at each change of the **Contact** record.*

Address1 - contains the **Contact** name of the record as the **Contact** name appears in the **Relationship Tree**. There is one critical addition to this statement. The name is entered into this field preceded by **2** spaces for each level of the tree. **Contact** records under a section are considered to be at the same level as the section. Let's look more closely at this. Specifically I will look at the information contained in this field for the record pertaining to **Consultants**, and **Bob Jefferson**. Both of these would be considered to be at level 2, therefore, in this field both **Consultants**, and **Bob Jefferson** will be proceeded by 4 spaces (2 spaces * 2 levels = 4 spaces).

City - this field is broken up into **3** concatenated sections. The first section is the eight character **UserID**. If the **UserID** is **DJ**, **2** characters, then this first section will be **DJ** plus six spaces. The next section contains the string representation of the date that this record was created. This might look like **20091027**. The next section, **7** characters long, is the creation time of this record. It is based on a **12** hour clock, and it includes a space if it is not a **2** digit hour. Here are a couple of possible representations of a value from this field:

DJ 20091027 9:31am
DJ 2009102710:49am

LastUser, **LastDate**, **LastTime**, and **RecID** - are all employed in the normal manner.

E-mail Address

Next I will examine the **ContSupp** table fields when the **RecType** is **P**. This single record type contains three subsets of possible record types. Let's start by assuming that the first record type that I am examining is one that is an **E-mail Address** for the main contact record. For this type of record most of the fields in this table are not employed.

Those fields that are employed, and their usage are:

AccountNo - stores the relational link back to the appropriate record in the **Contact1** table.

RecType - will always be **P** for this type of record.

Contact - will always contain **E-mail Address**, upgrades from previous versions might still have **Internet Address** as a value in this field.

ContSupRef - will contain the first **35** characters of the actual **E-mail Address**.

Notes - this text field will contain any characters that were entered into the **Notes** are of the dialog form for the **E-mail Address** at the time that said **E-mail Address** was created or after it was been modified.

Address1 - this field will contain any characters from the actual **E-mail Address** that exceeded the **35** character limitation of the **ContSupp.ContSupRef** field. As this field is **40** characters long, the longest **E-mail Address** that GoldMine is capable of handling is **75** characters.

Address2 - this field contains the **Contact** name associated with the **E-mail Address**.

City - this field is broken up into **3** concatenated sections. The first section is the eight character **UserID**. If the **UserID** is **DJ**, **2** characters, then this first section will be **DJ** plus six spaces. The next section contains the string representation of the date that this record was created. This might look like **20091027**. The next section, **7** characters long, is the creation time of this record. It is based on a **12** hour clock, and it includes a space if it is not a **2** digit hour. Here are a couple of possible representations of a value from this field:

```
DJ    20091027 9:31am
DJ    2009102710:49am
```

Zip - this is a flag field, and there are four possible flags. The first byte indicates whether this **E-mail Address** can use **Rich Text (HTML)**. The second byte indicates whether this is a **Primary E-mail Address** or not. In theory only one **E-mail Address** per **AccountNo** may have this flag set to **1**, indicating **Primary E-mail Address**. The third byte indicates the **MIME** capability of this **E-mail Address**, while the fourth, and the final byte, indicates a users decision to **Wrap Lines** or not.

MergeCodes - this field contains merge codes that are employed when doing an E-mail merge based on codes.

Status - I only mention this field here as it had relavence in previous versions of GoldMine. It remains **.null.** in this, the GoldMine Premium, version of GoldMine. In fact, of my **7814 E-mail Address** records, only **31** have a value in the **ContSupp.Status** field.

LastUser, **LastDate**, **LastTime**, and **RecID** - are all employed in the normal manner.

Tip

*Many users confuse the **ContSupp.MergeCodes** field with the **Contact1.MergeCodes** field. The two are separate and distinct, and have totally different functionality. For instance, one can create a **Filter** using the **Contact1.MergeCodes** field, whereas, this is not possible using the **ContSupp.MergeCodes** field.*

WebSite

As an addition to the last record type discussed, I will examine the fields when the RecType is still P, but this is an E-mail Address for an Additional Contact. All of the fields mentioned above pertain to this grouping with the addition of one field.

LinkAcct - this field contains the **RecID** from the linked ContSupp having a **RecType** of **C** (**Additional Contact**) record. This value will be something like:

```
4A7Y8PC,2+,$FJ:
```

Now let's examine the fields when the **RecType** is **P**, but this record type is a **Web Site** associated with the **Primary Contact**. The fields employed are:

AccountNo - stores the relational link back to the appropriate record in the **Contact1** table.

RecType - will always be **P** for this type of record.

Contact - will always contain the value **Web Site**

ContSupRef - this field will contain the URL to the web site.

Phone - could contain a **10** character link ID that is used in linking the **ContSupp H RecType** record, as well as for linking to lookup values in the **Lookup** table.

Notes - this text field will contain any characters that were entered into the **Notes** of the **Web Site** at the time that the **Web Site** was created or after it was edited. In addition, the **Notes** field takes

Note

*In some older versions of GoldMine, one used to be able to enter extra long **Web Site** addresses into the **Notes** field of the **ContSupp** record, and the **URL** could navigate to it from there. In GoldMine Premium, however, you are again allowed to enter exceedingly long **Web Sites**. Refer to the **Notes** definition.*

on special characteristics if the **Web Site** exceeds the **35** characters of the **ContSupp.ContSupRef** field.

You may expect it to look something like this:

~~REF=www.Long_Web_Site.com"~~NOTES=These are the actual Notes"

City - this field is broken up into **3** concatenated sections. The first section is the eight character **UserID**. If the **UserID** is **DJ**, **2** characters, then this first section will be **DJ** plus six spaces. The next section contains the string representation of the date that this record was created. This might look like **20091027**. The next section, **7** characters long, is the creation time of this record. It is based on a **12** hour clock, and it includes a space if it is not a **2** digit hour. Here are a couple of possible representations of a value from this field:

DJ 20091027 9:31am
DJ 2009102710:49am

Zip - this is a flag field, and there are four possible flags. The first byte is set to **0**. The second byte indicates whether this is a **Primary Web Site** or not. In theory only one **Web Site** per **AccountNo** may have this flag set to **1**, indicating the **Primary Web Site** for the related **Contact** record. The third byte, and the fourth byte is set to **0**. This field may or may not be populated.

Status - I only mention this field here as it had relavence in previous versions of GoldMine. It remains **.null.** in this, the GoldMine Premium, version of GoldMine. In fact, of my **7814 E-mail Address** records, only **31** have a value in the **ContSupp.Status** field.

LastUser, **LastDate**, **LastTime**, and **RecID** - are all employed in the normal manner.

Detail

Note

*All **Detail** records were formally known as **Profile** records in earlier versions of GoldMine. For legacy purposes, a **Detail** record maintains the **RecType** of **P** in all versions of GoldMine.*

Note

*As of GoldMine Premium 8.5.1.12 GoldMine, itself, exposes 12 **ContSupp** fields for **Extended Detail** field usage. This is better than the previous 8 that were being exposed within, but still not as good as **Details Plus** from **Solica** which exposes 13 fields, and additionally permits you to distinctly design a dialog form for each **Detail** that you define.*

And, finally, we examine the fields when the **RecType** is **P**, and this record is a default or user defined **Detail** type. The fields employed, and their usage are:

AccountNo - stores the relational link back to the appropriate record in the **Contact1** table.

RecType - will always be **P** for this type of record.

Contact - will contain the **Detail** name. i.e.:

Computer
Serial Number
Credit Card

ContSupRef - this field will contain the **Reference** information as supplied by the user creating this specific **Detail**, and, should the field contain a lot of repeatative entries, it is best to use the **F2 Lookup List** with the **Auto-Fill** option turned on.

Title - this field can be defined by the user as an extended field for this **Detail** within GoldMine under the **Info** tab.

Dear - this field can be defined by the user as an extended field for this **Detail** within GoldMine under the **Info** tab.

Phone - will contain a **10** character link ID that is used in linking the **ContSupp H RecType** record, as well as for linking to lookup values in the **Lookup** table.

Ext - this field can be defined by the user as an extended field for this **Detail** within GoldMine under the **Info** tab.

Fax - this field can be defined by the user as an extended field for this **Detail** within GoldMine under the **Info** tab.

LinkAcct - this field can be defined by the user as an extended field for this **Detail** within GoldMine under the **Info** tab.

Notes - this field contains any notes associated with this **Detail**.

Address1 - this field can be defined by the user as an extended field for this **Detail** within GoldMine under the **Info** tab.

Address2 - this field can be defined by the user as an extended field for this **Detail** within GoldMine under the **Info** tab.

Address3 - this field can be defined by the user as an extended field for this **Detail** within GoldMine under the **Info** tab.

City - this field is broken up into **3** concatenated sections. The first section is the eight character **UserID**. If the **UserID** is **DJ**, **2** characters, then this first section will be **DJ** plus six spaces. The next section contains the string representation of the date that this record was created. This might look like **20091027**. The next section, **7** characters long, is the creation time of this record. It is based on a **12** hour clock, and it includes a space if it is not a **2** digit hour. Here are a couple of possible representations of a value from this field:

DJ 20091027 9:31am
DJ 2009102710:49am

State - this field can be defined by the user as an extended field for this **Detail** within GoldMine under the **Info** tab.

Zip - this field can be defined by the user as an extended field for this **Detail** within GoldMine under the **Info** tab.

Country - this field can be defined by the user as an extended field for this **Detail** within GoldMine under the **Info** tab.

MergeCodes - this field can be defined by the user as an extended field for this **Detail** within GoldMine under the **Info** tab.

Status - the 1st byte could be either **0** or **1**, and for a **Detail** record other than **E-mail Address** or **Web Site**, would appear to have no relevance. While the 2nd byte, which could also be either **0** or **1**, indicates that there are not or are notes respectively in the **Notes** field.

LinkedDoc - this field is not used for this type of detail within GoldMine, but is exposed for usage by add-on products like **Details Plus** by **Solica**. This adds another memo field to the users repertoire when exposed by **Details Plus**.

LastUser, **LastDate**, **LastTime**, and **RecID** - are all employed in the normal manner.

Referral

The next to the last record type that I will examine for the **ContSupp** table is the **RecType** is **R**, and this record type is for a **Referral** type of record. One must remember that two records are created for each **Referral** created. The fields employed and their usage are:

Recommendation

*In my opinion, this type of **ContSupp** record is maintained for legacy purposes only. I always recommend that my clients do no use the **Referrals** tab, but, instead, remove it from view using the **User Override** as I discussed in Chapter 3 of this book.*

*I always recommend the creation of one **Contact** record for each contact, and that you set up any referential links using the **Relationship Tree**. The power of the **Relationship Tree** is usually under utilized by the end user. While the lack of power encompassed by the information under the **Referrals** tab is usually embraced by the end user.*

*As my wife would say:"**Go Figure**".*

AccountNo - stores the relational link back to the appropriate record in the **Contact1** table.

RecType - will always be **R** for this type of record.

Contact - Referral record A: **"For: "+ Contact1.Company**
- Referral record B: **"To: "+ Contact1.Company**

Title - Referral record A: **Contact1.AccountNo** of the referred **For:** company
- Referral record B: **Contact1.AccountNo** of the referred **To:** company

ContSupRef - this field will contain the Reference information as supplied by the user creating the Referral.

Ext - Referral record A: **R**
- Referral record B: **T**

LinkAcct - Referral record A: **Contact1.RecID** of the referred **For:** company
- Referral record B: **Contact1.RecID** of the referred **To:** company

Notes - if applicable, whatever notes were applied to both of the **Referral** records.

City - this field is broken up into **3** concatenated sections. The first section is the eight character **UserID**. If the **UserID** is **DJ**, **2** characters, then this first section will be **DJ** plus six spaces. The next section contains the string representation of the date that this record was created. This might look like **20091027**. The next section, **7** characters long, is the creation time of this record. It is based on a **12** hour clock, and it includes a space if it is not a **2** digit hour. Here are a couple of possible representations of a value from this field:

DJ 20091027 9:31am
DJ 2009102710:49am

State - this field is used, in this case, as a flags field. There are **13** possible flags. When the user creates a **Referral** there is a **Status** tab, see Figure 8-2, and on this tab are **13** options. Each byte in the **State** field represents one of these options. The **State** field should be identical for both **Referral** records.

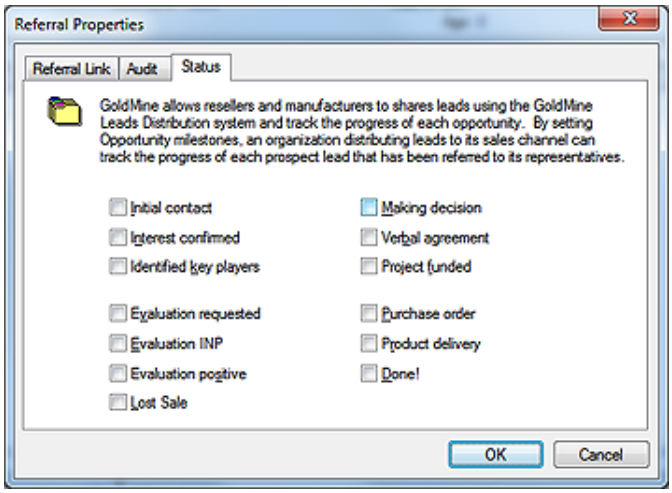


Figure 8-2

Status - this is a flag field as well. The 1st byte is set to **0**, but only if a note is added to trip the 2nd byte. The **0** is not removed from the 1st byte once it is established. When the 2nd byte is set to **1** this indicates that there are notes associated with these records.

LastUser, **LastDate**, **LastTime**, and **RecID** - are all employed in the normal manner.

Version

Now let's examine the fields when the **RecType** is **V** which appears to be a new **RecType** as of GoldMine Premium 8.5.1.12 or I simply missed it for my previous books. This record type appears to be a **Version** control, and there is but a single record for this **RecType**. The fields employed are:

AccountNo - contains a fixed value **CS_Version** which I can only assume stands for **ContSupp Version**. That does seem a reasonable assumption, does it not?

RecType - will always be **V** for this type of record.

Contact - will always contain the version number which, in my case, was **8.10.90514**.

All of the rest of the **ContSupp** fields are either **.null.** or empty except for **U_Contact** and, of course the **RecID** which can never be **.null.** or empty.

ContGrps

ContGrps Table

Indexes

GROUPACC	AccountNo, UserID (Non-Unique, Non-Clustered)
GROUPNO	UserID, U_Code, RecID (Non-Unique, Non-Clustered)
GRPRECID	RecID - Unique (Unique, Non-Clustered)

Relationships

ContGrps.Accountno	Many-to-One	**Contact1.AccountNo**

Structure

AccountNo	VarChar	20	Linked Contact AccountNo - not Null
Code	VarChar	8	Code Assigned to the Group
RecID	VarChar	15	Record ID - not Null
Ref	VarChar	24	Reference
U_Code	VarChar	8	Upper Code - not Null
UserID	VarChar	15	User - not Null

The **CONTGRPS** table stores the names of groups, and the individual group members, so records have two formats.

For **Group Header** records the contents are:

UserID – the **UserID** of the owner of the **Group** or blank for a **(public) Group**.

Code - the code given to the group at the time of creation or when modifying a **Group**.

AccountNo – for **Header** records is always ***M** or ***MS** followed by the number of members in the group, right justified to utilize the full **20** character space. The **S**, if present, shows this is a group that is permitted to synchronize. For Example: a value of ***M 282** shows a non-synchronizing group possessing 282 members.

Ref – the name of the **Group** assigned at the time of the creation of the **Group**.

RecID – normal meaning.

For **Group Member** records the contents are:

UserID – the **RecID** of the associated **Group Header** record. It is **ContGrps.UserID** which links all the members to a particular **Group**.

Code – the sort order of the **Group** as selected (1st eight characters) when the **Group** was created or the **Member** added.

AccountNo – the **Accountno** of the **Contact** record of the **Group Member**.

Ref – the **Reference** of the **Group** as selected when the **Group** was created or the **Member** added.

It is helpful to know the **ContGrps** structure when writing SQL queries. For example the query:

```
select *
from ContGrps
where UserID='DJ'
  and Ref = 'Customers'
```

Could return the **Group Header** record for this group as:

Row	Userid	Code	Accountno		Ref	Recid
1	DJ	DJH	*M	3024	Customers	CIN12R9)<I+IR#Y

where as this query would show the individual members:

```
select *
from ContGrps
where UserID in (select RecID
      from ContGrps
      where UserID='DJ'
      and Ref = 'Customers')
```

This could be useful for more sophisticated queries, for example combining group membership with other criteria.

Note

*Occasionally the **Group Header** record could possess the wrong count. This can be fixed by cloning the group. The new group should have the correct count. Cloning is also useful to change a Non-Syncing Group into a Syncing Group and vice versa.*

Cal

CAL Table

Indexes

CAL	RecType, UserID, OnDate, OnTime, RecID (Non-Unique, Non-Clustered)
CALALARM	AlarmFlag, UserID, AlarmDate, AlarmTime (Non-Unique, Non-Clustered)
CALCDCINST	CalDef_ID, AccountNo, Is_Exception, Orig_Date, Orig_Time (Non-Unique, Non-Clustered)
CALCDEX	CalDefEx_ID (Non-Unique, Non-Clustered)
CALCDINST	CalDef_ID, Orig_Date, Orig_Time (Non-Unique, Non-Clustered)
CALCDUINST	CalDef_ID, UserID, Is_Exception, Orig_Date, Orig_Time (Non-Unique, Non-Clustered)
CALCONT	AccountNo, RecType, OnDate, OnTime, RecID (Non-Unique, Non-Clustered)
CALDATE	UserID, OnDate, OnTime, RecID (Non-Unique, Non-Clustered)
CALENDDATE	EndDate (Non-Unique, Non-Clustered)
CALLINKRECID	LinkRecID (Non-Unique, Non-Clustered)
CALPROB	RecType, UserID (Non-Unique, Non-Clustered)
CALRECID	RecID (Unique, Non-Clustered)
CALRLINK	LOPRecID, RecType, OnDate, OnTime (Non-Unique, Non-Clustered)

Relationships

Cal.AccountNo	Many-to-One	**Contact1.AccountNo**
Cal.RecID	One-to-One	**MailBox.LinkRecID**
Cal.LinkRecID	One-to-One	**MailBox.RecID**
Cal.LOpRecID	Many-to-One	**OpMgr.RecID**
Cal.LOpRecID	Many-to-One	**Cases.RecID**
Cal.Def_ID	One-to-One	**CalDef.RecID**

Structure

AccountNo	VarChar	20	Linked Contact AccountNo - not Null
AConfirm	VarChar	3	Meeting Confirmation Flag
ActvCode	VarChar	3	Activity Code
AlarmDate	DateTime	8	Alarm on Date
AlarmFlag	VarChar	1	Alarm Flag - not Null
AlarmTime	VarChar	5	Alarm on Time - not Null
ApptUser	VarChar	10	Meeting Confirmation User/Color Flag Byte 10
Attendees_Ex	VarChar	100	Refer to sidebar Note
CalDef_ID	VarChar	15	Refer to sidebar Note
CalDefEx_ID	VarChar	15	Refer to sidebar Note
Company	VarChar	60	Company/Contact Name associated w/activity
CreateAt	VarChar	5	Created at Time
CreateBy	VarChar	8	Creation User
CreateOn	DateTime	8	Created on Date
DirCode	VarChar	10	DirCode for the Contact Set
Duration	SmallInt	2	Duration/Probability
EndDate	DateTime	8	Activity Scheduled End Date
Ext	VarChar	5	Used to code format (**html** or **plain**)
Flags	Ineger	16	Refer to sidebar Note
Is_Exception	SmallInt	2	Refer to sidebar Note
LastDate	DateTime	8	Last modified on date
LastTime	VarChar	5	Last modified at time
LastUser	VarChar	8	Last modified by user
LDocRecID	VarChar	15	Reserved for possible future use
LinkRecID	VarChar	15	Linked Record ID - not Null
LOpRecID	VarChar	15	Linked Opportunity Record ID - not Null
Notes	Image	16	Notes
Number1	Float	8	Sales Potential Amount
Number2	Float	8	Unit in a Forecasted Sale
OnDate	DateTime	8	Activity on Date
OnTime	VarChar	5	Activity on Time - not Null
Orig_Date	DateTime	8	Refer to sidebar Note
Orig_Time	VarChar	5	Refer to sidebar Note
RecID	VarChar	15	Record ID - not Null
RecType	VarChar	1	Record Type - not Null*
Ref	VarChar	80	Reference
RSVP	VarChar	1	RSVP Notification Flag
Service	Text	16	Notes
Status	VarChar	4	GoldMine field**
UserID	VarChar	8	User - not Null

Note

In a sampling of 500 Calendar Activities from my production database these fields were not utilized as of this writing:

Cal.Attendees_Ex
CalDef_ID
CalDefEx_ID
Cal.Flags
Cal.Is_Exception
Cal.Orig_Date
Cal.Orig_Time

Note

*Please notice that **Cal.Notes** is now an **Image** based field, and in the normal SQL Query, unreadable. Use this as part of your select statement to make these notes readable:*

```
select cast(cast(NOTES as varbinary(max)) as varchar(max)) as Notes
```

*** The following are possible values for the RecType field:**

A	Appointment	**E**	Event	**O**	Other Activity
B	Occasions	**F**	Literature Fulfillment	**Q**	Queued E-mail/Quota
C	Phone call	**H**	Holidays	**S**	Sales Potential
D	To-do	**M**	Message	**T**	Next Action

**** Status**

1st character **Flag**
2nd character **1** if there are notes in the **Notes** field.

WARNING

*The accuracy of the **Peg Board** is dubious, and should never be relied upon for user accountability. If GoldMine is closed improperly or closed using the **X**, then the **Peg Board** is not updated. There are many other instances that could cause incorrect data in the **Peg Board**.*

This table holds all of the **Pending** activities for **Contacts**, and **Users** alike. **Cal** table records can be broken up by **RecType** as we have seen in the other GoldMine tables. The same holds true for this table as in the **ContHist** table, activities with the **AccountNo** field populated are linked activities. Some of the activities with nothing in the **AccountNo** field are unlinked activities, while others are there by default. For example, there are a number of records with an **AccountNo** like **PB:DJ**, **PB:MASTER**, etc. These records are the **Peg Board** (**PB:**) records for the various users. It is the information from these records that is utilized in the **Peg Board** tab display on the GoldMine graphical calendar.

In this type of record the **Cal.Notes** field for a given user maintains the users to be displayed on the **Peg Board**, while **Cal.Company** field for each **PB:UserID** holds the monitored information. This information is from the **PB:DJ** record in my **Cal** table:

```
Cal.Company = 20091009 09:54 20091009 09:54 20091009 10:03 20091009 10:04
Cal.Notes = DJ;MASTER;
```

You will notice that I am monitoring the GoldMine **Peg Board** activity for the **DJ**, and the **MASTER** logins.

Appointments, Calls, Other Actions, Next Actions

I will examine four of the **RecType** values at the same time as they all use the same dialog form for the input of the information. A typical input dialog form is displayed below in Figure 8-3 to **Schedule an Appointment**.

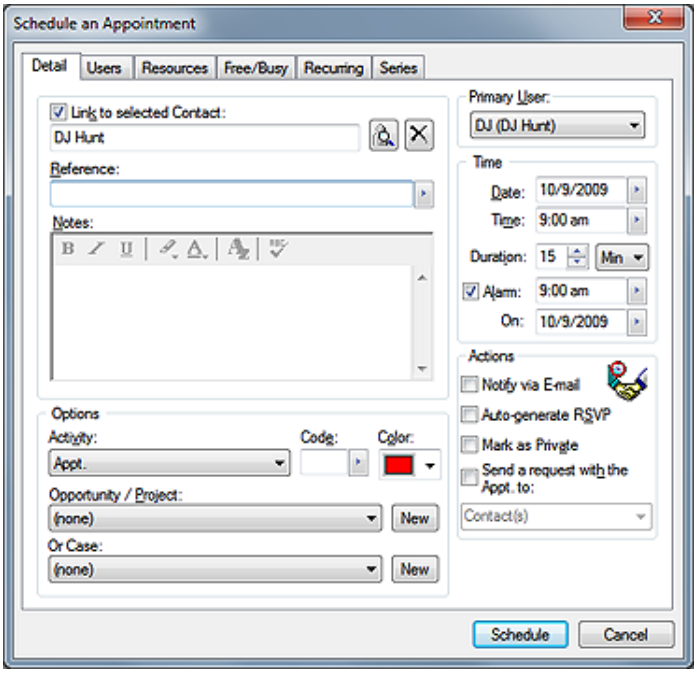


Figure 8-3

Note

*This has been asked many times over, so I will answer it for you here, the **Activity:** field contains a hardcoded (fixed) list to which you may neither add to nor remove from list items. This field is the string representation of the **RecType** for the particular activity being scheduled.*

The **Activity:** field is the only thing that needs to be changed to schedule a different **RecType** (**A**, **C**, **O**, and **T**). Let's begin by looking at the various fields in a record as they pertain to these record types.

UserID - (**Primary User:**) stores the GoldMine **UserID** for whom this activity is scheduled.

AccountNo - stores the relational link back to the appropriate record in the **Contact1** table if the ☑ **Link to selected Contact**. option remains selected which is it's default state.

OnDate (**Date:**) - contains the date for which this activity is to be scheduled.

Tip

*The **OnTime** field can accept a character value such as **A**. By entering such the activity will be placed on the **Task** portion on your graphical calendar. It may still be a dated, and even an alarmed item on your graphical calendars **Task** listing unlike **To-Do's**.*

Note

*The various color coding values for the 10th byte of **ApptUser** are:*

***Blue** (default) = space or null*
***Magenta** = !*
***Red** = " (double quote)*
***Cyan** = #*
***Green** = $*
***Yellow** = %*
***Dark Cyan** = &*
***White** = ' (single quote)*
***Light Gray** = (*
***Maroon** =)*
***Dark Green** = **
***Dark Yellow** = +*
***Dark Blue** = , (comma)*
***Purple** = -*
***Dark Gray** = . (period)*
***Black** = /*

WARNING

***Cal.Notes** will contain **HTML** coding, and this coding will be visible when doing reports unless you, the GoldMine Administrator, specifically turn off this questionable feature or accommodate it in your reports.*

You must add:

HTML_Cal_Notes = 0

*in the **GM.ini** under the **[GoldMine]** section in order to turn off **HTML** coding. Alternatively, you may use the GUI:*

Tools
Configure ►
System Settings
***Display** tab*
***Notes Format** frame*

WARNING

*Programmers please note that the **LOpRecID** field must be populated. If there is no linked **Opportunity** or **Case**, then this field **must** have a space as the 1st character followed by an API generated **RecID**. This requirement was included to help with indexing, hence performance.*

OnTime (**Time:**) - contains the time, in a 24 hour format (**i.e. 2:00pm = 14:00**) when entered programmatically, for which the activity is scheduled.

EndDate - in all cases of these **RecTypes** that I have examined, this field appears to hold the same date/time as the **OnDate** field. I have even tried scheduling recurring appointments, and, as each activity is schedule separately, each has the same **OnDate**, and **EndDate**.

AlarmFlag - this field must always be populated with either an **N** for no alarm, or a **Y** if the ☐ **Alarm:** option has been selected.

AlarmDate (**Alarm:**) - contains the date on which an alarm is to be triggered, if any.

AlarmTime (**On:**) - contains the time an alarm is to be triggered, if any. Again, this time is entered into the table in a 24 hour format (**i.e. 2:00pm = 14:00**) when it is entered programmatically.

ActvCode (**Code:**) - could contain up to three characters which act as an activity code for the given activity. This helps to granularize your GoldMine data for more focused reporting and queries.

RSVP - is a one character field that must contain an **N** if the ☐ **Auto-generate RSVP** option has not been selected in the scheduling dialog form, and a **Y** if this option has been selected.

Duration (**Duration:**) - when scheduled through the GoldMine GUI, **Duration:** may be scheduled in minutes or hours. The resulting value, stored in the **Duration** field of the record, is always stored in minutes. (**i.e. 16.2 hrs = 972**)

RecType - for this section of the chapter is, obviously I hope, either **A**, **C**, **O** or **T** depending upon the type of activity being scheduled.

AConfirm - despite what the structure table mentions, I have not found this to be used anywhere in my **Cal** table. In fact, after scheduling activities this field contains a **.null.** value.

ApptUser - despite what the structure table mentions, I have only seen that the 10th byte is used, and it is used to store the character representing the color to be displayed for the activity when the activity is being displayed in the GoldMine graphical calendar.

Status - the 1st byte in this field may contain a **B**, if this activity is one of multiple recurring activities that were scheduled at the same time. The 2nd byte will be either a **0** or a **1** if there are no notes, or if there are notes, respectively. The 2nd byte should always contain a value.

DirCode - is, quite simply, the **File Code** for the contact dataset for which this activity has been scheduled against. In the past, this field was not employed. Today, the information in this field has some significance and, hence, should be populated. The **DirCode** will be found in the **SpFiles** table.

Number1 - for these rectypes, this field is used to identify whether this activity has been marked by the creator as **Private**, ☐ **Mark as Private**. If the activity has been designated as **Private** then this field will contain **16** otherwise the field will contain a **0**.

Number2 - I have not found any indication of this field being employed for any of these **RecTypes**, hence, being float type of field, a **0** will be contained in this field.

Company - this field is used to store the **Contact** name, or **Company** name when the **Contact** field is blank, of the contact record against whom the activity was scheduled (linked).

Ref (**Reference:**) - is up to 80 characters of a short description for this scheduled activity.

Notes (**Notes:**) - is employed to hold any text type **Notes** that were entered for this activity. Be mindful, in GoldMine Premium, these notes may be in HTML format, unless that option was specifically turned off in the GM.ini (see sidebar). Additionally, as of GoldMine Premium 8.5.1.12 this field is a SQL Image based field as opposed to the former SQL Text based field.

LinkRecID - will maintain the **RecID** from the **MailBox** record, if the creator of the activity has selected the option to ☐ **Send a request with the <<Activity Name>> to:**. Once entered, this information cannot be removed through GoldMine. It may only be removed through the use of an external application.

LDocRecID - for these **RecTypes**, I have never observed that this field was populated, and, in fact, is populated with a **.null.** value for these **RecTypes**.

LOpRecID - is employed when the activity is associated with an **Opportunity**/**Project** or a **Case** and, in those situations, this field would contain the **RecID** of said **Opportunity**/**Project** or **Case** as

Note

*As the **LOpRecID** field is but one field, it should be obvious that one can never have an **Opportunity**/**Project** and a **Case** linked to the same activity.*

recorded in the **OpMgr** table or the **Cases** table that the relationship is maintained against. Refer to Note in the sidebar.

If there is no relationship with an **Opportunity**/**Project** or a **Case** then the application must populate this field with a **space** followed by a generated **RecID** (i.e. **space(1)+LNC8TK#>(+< W<**). This bit of information is probably only relavent to you readers who are developing external applications for use against the GoldMine tables.

CreateBy, **CreateOn**, **CreateAt**, **LastUser**, **LastDate**, **LastTime**, **Ext**, **Service**, and **RecID** - are all employed in the normal manner.

Occasion

Cal.Rectype = B is used for displaying an **Occasion** on the GoldMine graphical calendar. Figure 8-4 and Figure 8-5 on the next page, show the **Edit an Occasion** dialog form. This set of dialogs is accessed via menu selection of:

Edit
Record Properties ►
Contact Details...
Occasions tab

The record results from these screen entries, and/or if the record were to be added by an external application are as follows:

UserID (**User:**) - stores GoldMine **UserID** login for the user creating this record or the selected **User** if modified.

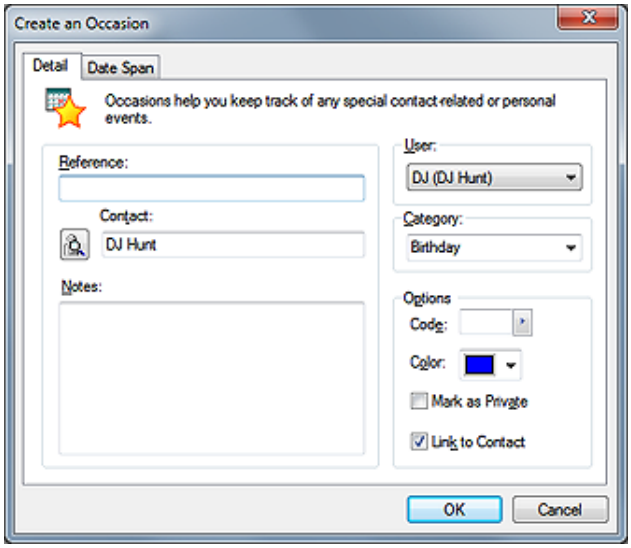


Figure 8-4

AccountNo - stores the relational link back to the appropriate record in the **Contact1** table if the ☑ **Link to Contact** option is selected. If ☑ **Link to Contact** is not selected, or if an external application creating the record does not want to link the occasion to a contact record, then this field would remain blank.

OnDate (**When** frame **Date:**) - is the date on which the first occurrence of the occasion happens.

OnTime - occasions are not displayed in the timed section of the graphical calendar, and, therefore, do not make use of this field.

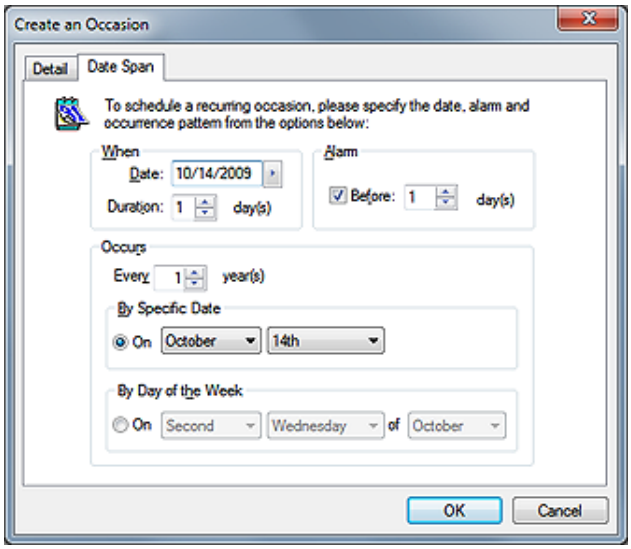


Figure 8-5

EndDate - this date is based on the **Duration:** information that was entered. In Figure 8-5, you can see that we entered a **Duration:** of **1 day(s)**, therefore, the **EndDate** field would be **1 day** from the **OnDate** of **10/14/2009** or **10/15/2009**.

AlarmFlag (**Alarm** frame) - this field must always be populated. For this **RecType** there is no graphical option, but a programmer could populate this field through their application with a **Y** or an **N**. This option is simply displayed in this frame as a ☑ which is selected in its default state. There is no label for this option beyond the frame label **Alarm**.

AlarmTime - if an alarm has been set, the **AlarmTime** is always **00:00** unless programmatically altered.

AlarmDate (**Alarm** frame **Before: X day(s)**) - if an alarm is set, it may be set to occur from **0** to any number of days before the established **OnDate**. GoldMine will calculate the date for this field based on the number set by the user in **Before: X day(s)**. External applications just need to enter a date for the alarm in this field.

ActvCode (**Code:**) - could contain as many as three characters which act as an activity code for the given activity. This helps to granularize your GoldMine data for more focused reporting.

RSVP - is not populated for this record type, and actually remains as a **.null.** value.

Duration - when scheduled through the GoldMine GUI, **Duration:** may be scheduled only in days. External applications simply need to populate the number of days over which the occasion is to run. See related **EndDate** field.

RecType - for this section of the chapter is **B**.

AConfirm - despite what the structure table mentions, we have not found this to be used anywhere in our **Cal** table.

ApptUser (**Color:**) - despite what the structure table mentions, we have only seen that the 10th byte is used to store the character representing the color to be used for the activity when the activity is displayed in the graphical calendar. In Figure 8-4, if I had selected Magenta, this field would contain:

[!]

Status - this field is not used for this record type even if there are notes associated with the record. This is because **RecType B** always has notes as we'll show you in a bit. The field is set to the default **.null.** value.

DirCode - is, quite simply, the **File Code** for the contact dataset for which this activity has been scheduled against. In the past, this field was not employed. Today, the information in this field has greater significance, hence should be populated. The **DirCode** will be found in the **SpFiles** table or through the GUI:

Tools
Databases
Open Databases...

The **File Code** column can be seen in the **Contact Set Databases** dialog form as seen here in Figure 8-6. You will notice that the **DirCode** for my database will be **COMMON**.

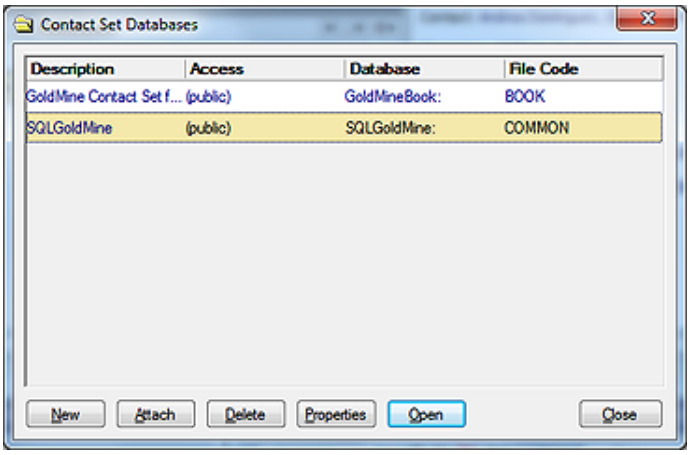


Figure 8-6

Number1 - for these **RecTypes**, this field is used to identify whether this activity has been marked by the creator as **Private**. If the activity has been designated as **Private** then this field will contain **16**, otherwise **0**.

Number2 (**Category:**) - depending on the **Category:** selected this number could be anything from a **0** to **5**. In Example:

0 = Other
1 = Anniversay
2 = Birthday
3 = Festival
4 = Reunion
5 = Retreat

Company - this field is used to store the **Contact** name, or **Company** name, when the **Contact** field is blank, of the contact record for whom the activity was scheduled against.

Note

*The various color coding values for the 10th byte of **ApptUser** are:*

***Blue** (default) = space or null*
Magenta = !
***Red** = " (double quote)*
Cyan = #
Green = $
Yellow = %
Dark Cyan = &
***White** = ' (single quote)*
Light Gray = (
Maroon =)
Dark Green = *
Dark Yellow = +
***Dark Blue = ,** (comma)*
***Purple** = -*
***Dark Gray = .** (period)*
Black = /

Ref (**Reference:**) - is up to 80 characters of a short description for this scheduled occasion.

Notes (**Notes:**) - is employed to hold any memo type notes in this image type field that were entered for this activity. Be mindful, in GoldMine Premium, these notes may be in HTML format, unless that option was specifically turned off in the GM.ini as I discussed in an earlier chapter. The notes field for this record type contain very specific information. A typical setting might be:

```
BEGIN:VCALENDAR
 PRODID:-//FrontRange Solutions//GoldMine 8.0//EN
 VERSION:2.0
 METHOD:PUBLISH
 BEGIN:VEVENT
  ORGANIZER;CN="DJ (DJ Hunt)":MAILTO:--X-GM-USER-- <DJ>
  DTSTART;VALUE=DATE:19700902
  DTEND;VALUE=DATE:19700903
  RRULE:FREQ=YEARLY;COUNT=30;INTERVAL=1;BYMONTHDAY=2;BYMONTH=9;WKST=SU
  TRANSP:OPAQUE
  SEQUENCE:0
  UID:4543465944334520404F2934522359
  DTSTAMP:20091016T131446Z
  SUMMARY:Andreas Birthday
  PRIORITY:5
  CLASS:PRIVATE
  CATEGORIES:Birthday
  BEGIN:VALARM
   TRIGGER:-P1D
   ACTION:DISPLAY
   DESCRIPTION:Reminder
  END:VALARM
 END:VEVENT
END:VCALENDAR
```

Note

*Indentation is for book presentation only. In the **Notes** field, no such formatting is encountered.*

LinkRecID, **LDocRecID** - are not populated for this record type.

LOpRecID - for this record type there is no association with an **Opportunity**/**Project**, any external application must populate this field with a space followed by a generated **RecID** (i.e. **space(1)+9F6VPQ2%+IX]R#**).

WARNING

*Programmers please note that the **LOpRecID** field must be populated. If there is no linked opportunity, then this field **must** have a space as the 1st character followed by an API generated **RecID**. This requirement was included to help with indexing, hence performance.*

CreateBy, **CreateOn**, **CreateAt**, **Ext**, **Service**, and **RecID** - are all employed in the normal manner.

LastUser, **LastDate**, and **LastTime** - are not populated for this record type.

To-Do

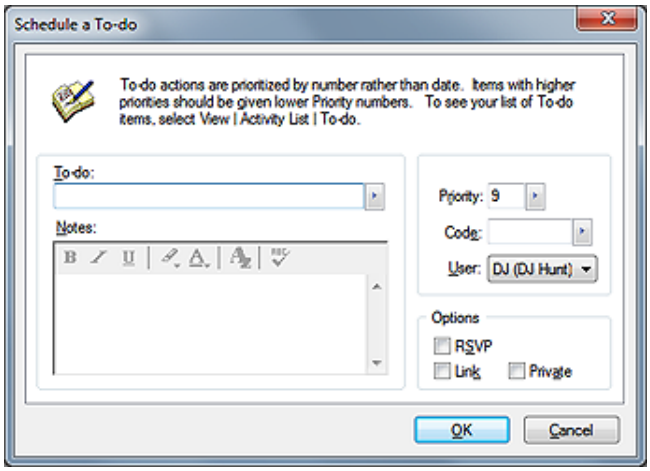


Figure 8-7

Cal.Rectype = D is used for displaying untimed **Tasks**, in GoldMine called a **To-Do**, on the graphical calendar. As one can see in Figure 8-7, not as much information is required to **Schedule a To-do**. They are untimed activities such that no clock is required, and no alarms are available via the GUI. (see **TIP** for **Appointments, Calls, Other Activities, and Next Actions**)

UserID (**User:**) - maintains the GoldMine **UserID** for whom the **To-Do** is scheduled.

AccountNo - stores the relational link back to the appropriate record in the **Contact1** table. If ☐ **Link** is not selected, or if an external application creating the record does not want to link the task to a contact record, then this field would be left blank. Unlike the **Appointment**, **Call**, **Other Action** and **Next Action**, this option is not selected in the default state.

OnDate - for a task this field always remains **.null.** for this **RecType**. **To-Dos** are displayed everyday on the graphical calendar as long as they have not been completed.

Note

*Programmers should be aware that in the **OnTime** field, one is not limited to values between **1** and **9**. You could as easily use **A** thru **Z**.*

OnTime (**Priority:**) - maintains the **Priority:** assigned to this task either by the user or programmatically by a developer. Priorities are assigned **1** through **9** normally, **1** being the highest, and **9** being the lowest priority. This information is stored in the first byte of the **OnTime** field.

Note

AlarmDate *and* ***AlarmTime*** *may be set programmatically, and these settings will be honored by the GoldMine* ***Calendar****. Additionally, within GoldMine, specifically the* ***UserID.ini****, or with an override in the* ***GM.ini****, the* ***OnByDefault=D*** *may be set to have* ***To-Dos*** *created in GoldMine alarmed by default.*

Note

Even though one could type many characters into the ***ActvCode*** *field, when saved, only the 1st 3 characters will be retained.*

EndDate - is populated with the creation date of the scheduled task.

AlarmFlag - for a scheduled task, will always contain an **N** if the **To-Do** was created via GoldMine. If, however, the **To-Do** was created programmatically then this flag could be assigned a **Y**, and the alarms would be activated.

AlarmDate, **AlarmTime** - are not populated as there is no alarm that can be set unless you have set up your **UserID.ini** or **GM.ini** to alarm this **RecType** in which case the **AlarmDate** field will contain the creation date for the activity. The **AlarmTime** will contain **00:00** (midnight).

ActvCode (**Code:**) - could contain up to three characters which act as an activity code, identified as **Code:** in the dialog form as shown in Figure 8-7 above. This helps to granularize your GoldMine data for more focused reporting.

RSVP (**RSVP**) - is a one character field that must contain an **N** if the ☐ **RSVP** option has not been selected in the scheduling dialog form, and a **Y** if this option has been selected.

Duration - for the **To-Do RecType** should always be a **.null.** value.

RecType - for this section of the chapter is **D**.

AConfirm - despite what the structure table mentions, we have not found this to be used anywhere in our **Cal** table, and for this **RecType** contains a **.null.** value.

ApptUser - color coding on the graphical calendar for a task is not employed within GoldMine, hence this field is always empty for this **RecType**.

Status - the 2nd byte will be either a **0** or a **1** if there are no notes or if there are notes, respectively. The 2nd byte must always contain a value.

DirCode - is, quite simply, the **File Code** for the contact dataset for which this activity has been scheduled against. In the past, this field was not employed. Today, the information in this field has greater significance, hence should be populated. The **DirCode** will be found in the **SpFiles** table or through the GUI:

Tools
Databases
Open Databases...

Number1 - for this **RecType**, this field is used to identify whether this activity has been marked by the creator as **Private**. If the activity has been designated as **Private** then this field will contain the number **16** otherwise the number **0**.

Number2 - for the **To-Do RecType** will always be **0**. As this is a **Float** based field, the field will always contain a **0**, and there is no need for an application developer to populate this field externally.

Company - this field would normally be used to store the **Contact** name (**Company** name when the **Contact** field is blank), however, even when linked, in GoldMine 8.5.1.12, this field remains empty (not **.null.**, but **empty**).

Ref (**To-do:**) - contains up to 80 characters of a short description for this scheduled activity.

Notes (**Notes:**) - is employed to hold any text type **Notes**, in an image based field, that were entered for this activity. Be mindful, in GoldMine Premium, these notes may be in HTML format unless that option was specifically turned off in the **GM.ini** as I discussed earilier in this book.

LinkRecID - will remain empty for this record type.

WARNING

Programmers please note that the ***LOpRecID*** *field must be populated. If there is no linked opportunity, then this field* ***must*** *have a space as the 1st character followed by an API generated* ***RecID****. This requirement was included to help with indexing, hence performance.*

LDocRecID - will remain contain a **.null.** value for this record type.

LOpRecID - is employed when the activity is associated with an **Opportunity**/**Project**/**Case**, and, as this **RecType** can not be associated with one of these, this field must contain a generated value. An external application must populate this field with a space followed by a generated **RecID** (i.e. **space(1)+9F6VPQ2%+IX]R#**).

CreateBy, **CreateOn**, **CreateAt**, **LastUser**, **LastDate**, **LastTime**, **Ext**, **Service**, and **RecID** - are all employed in the normal manner.

Event

Cal.Rectype = E is used for displaying **Events** on the GoldMine graphical calendar. As one can see in Figure 8-8 on the next page, scheduling **Events** is not that much different from scheduling an **Appointment**, **Next Action**, **Other Action**, etc. These are untimed events such that no clock is required but, in this case, alarms are available via the GUI. **Duration:** can only be given in **x Day**

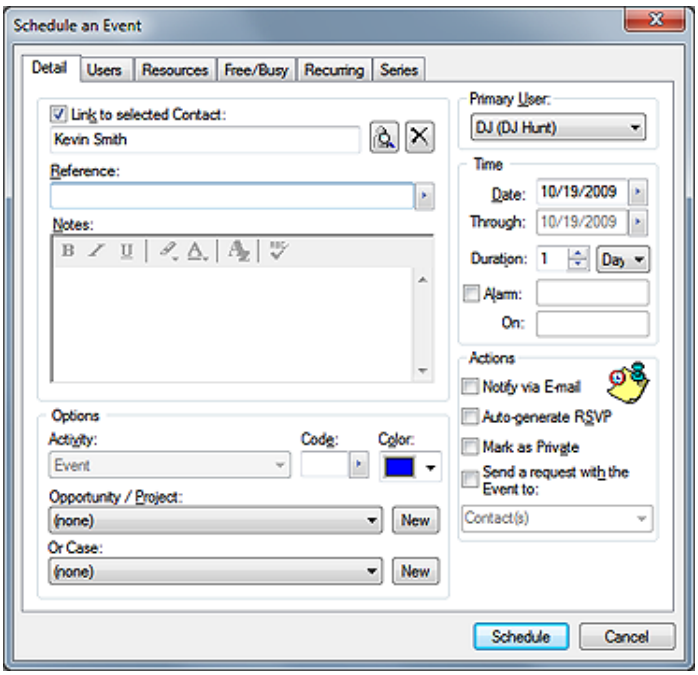


Figure 8-8

(sorry, but that is what the dialog form says) even though it is a drop list and would appear to contain more options. **Events**, as they are untimed, are displayed in the graphical calendar under **Task** listing along with all the other untimed activities such as **To-Do's**. Even though there is a **Through:** field displayed on the **Schedule an Event** GUI, this value is controlled programmatically based on the **Duration:** that one enters.

UserID (**Primary User:**) - maintains the GoldMine **UserID** against whom the event is being scheduled.

AccountNo - stores the relational link back to the appropriate record in the **Contact1** table. If ☑ **Link to selected Contact:** is not selected, or if an external application creating the record does not want to link the **Event** to a contact record, then this field would be left blank. Like an **Appointment**, **Call**, **Other Action** and **Next Action**, this option is selected in the default state.

OnDate (**Date:**) - stores the date on which the event begins. Conflict checking is only performed through the GoldMine interface.

OnTime - remembering that **Events** are not timed activities, this field should always remain **empty** for this **RecType**.

EndDate (**Through:**) - for a change is populated appropriately. If the **Duration:** were populated with **3 Days**, then the **EndDate** would be **OnDate** plus 2 days.

AlarmFlag - this field must always be populated with either an **N** for no alarm, or a **Y** if the ☐ **Alarm:** option has been selected.

AlarmTime (**On:**) - contains the time an alarm is to be triggered, if any. Again, this time is entered into the table in a 24 hour format (i.e. **15:50**).

AlarmDate (**Alarm:**) - contains the date on which an alarm is to be triggered, if any, and is usually the same date as the **OnDate** value.

ActvCode (**Code:**) - could contain up to three characters which act as an activity code. This helps to granularize your GoldMine data for more focused reporting.

RSVP - is a one character field that must contain an **N** if the ☐ **Auto-generate RSVP** option has not been selected in the scheduling dialog form, and a **Y** if this option has been selected.

Duration (**Duration:**) - will maintain, in whole days, the length of the event, and should not be converted to minutes.

RecType - for this section of the chapter is **E**.

Note

*Programmers are reminded that **empty** and **.null.** Are separate and distinct values.*

Note

*The various color coding values for the 10th byte of **ApptUser** are:*

***Blue** (default) = space or null*
Magenta = !
***Red = "** (double quote)*
Cyan = #
Green = $
Yellow = %
Dark Cyan = &
***White = '** (single quote)*
Light Gray = (
Maroon =)
Dark Green = *
Dark Yellow = +
***Dark Blue = ,** (comma)*
Purple = -
***Dark Gray = .** (period)*
Black = /

AConfirm - despite what the structure table mentions, we have not found this to be used anywhere in our **Cal** table.

ApptUser (**Color:**) - color coding on the graphical calendar for an **Event** is not employed by GoldMine, hence this field could be left empty for this record type. When scheduling an **Event**, however, you are permitted to select a color. Should the user select a color the code for that color will be stored in the 10th byte of this field.

Status - the 2nd byte will be either a **0** or a **1** if there are no notes, or if there are notes, respectively. The 2nd byte must always contain a value.

DirCode - is, quite simply, the **File Code** for the contact dataset for which this activity has been scheduled against. In the past, this field was not employed. Today, the information in this field has greater significance, hence should be populated. The **DirCode** will be found in the **SpFiles** table or through the GUI:

Tools
Databases
Open Databases...

Number1 - for this **RecType**, this field is used to identify whether this activity has had ☐ **Mark as Private** selected by the creator. If the activity has been designated as **Private**, then this field will contain **16** otherwise **0**.

Number2 - for the **Event** will always be **0**. As this is a **Float** based field, the field will always contain a **0**, and there is no need for an application developer to populate this field externally.

Company - this field is used to store the **Contact** name, or **Company** name when the **Contact** field is blank, of the contact record for whom the activity was scheduled against, if the **Event** is linked, as it is in the default state, using the ☑ **Link to selected Contact:** option. Otherwise, this field will remain empty.

Ref (**Reference:**) - is up to 80 characters of a short description for this scheduled activity.

Notes (**Notes:**) - is employed to hold any text type **Notes**, in an image based field, that were entered for this activity. Be mindful, in GoldMine Premium, these notes may be in HTML format unless that option was specifically turned off in the **GM.ini** as I discussed earilier in this book.

LinkRecID - will maintain the **RecID** from the **MailBox** item, if the creator of the activity has selected the option to **Send a request with the Event to:**. Once entered, this information cannot be removed through GoldMine. It may only be removed through the use of an external application.

WARNING

*Programmers please note that the **LOpRecID** field must be populated. If there is no linked opportunity, then this field **must** have a space as the 1st character followed by an API generated **RecID**. This requirement was included to help with indexing, hence performance.*

LDocRecID - for this **RecType**, we have never seen this field populated, and, in fact, is **.null.**.

LOpRecID - is employed when the activity is associated with an **Opportunity**/**Project**/**Case**, and, if there is such an association, this field must contain the **RecID** of the **Opportunity**/**Project** record or the **Case** record. As this is only one field an **Event** can only be associated with either an **Opportunity**, a **Project** or a **Case**. One cannot associate an **Opportunity** and a **Case** for instance. An external application must populate this field with a space followed by a generated **RecID** if there is no association (i.e. **space(1)+9F6VPQ2%+IX]R#**).

CreateBy, **CreateOn**, **CreateAt**, **LastUser**, **LastDate**, **LastTime**, **Ext**, **Service**, and **RecID** - are all employee in the normal manner.

Literature Fulfillment

Cal.Rectype = F is used for displaying **Literature Fulfillment Requests** on the graphical calendar. **Literature Requests** are dated, but untimed activities. Hence, they appear on the graphical calendar in the untimed **Task** list on the day for which they are scheduled. Figure 8-9, on the next page, shows the input dialog form for this **RecType**.

UserID (**Assigned to:**) - maintains the GoldMine **UserID** to whom the **Literature Fulfillment Request** activity has been delegated.

AccountNo - stores the relational link back to the appropriate record in the **Contact1** table. There is no link option associated with this activity as literature requests are always linked to a contact record.

Note

*Programmers must understand the GoldMine paradigm for the **OnTime** and the **AlarmTime** fields, and be careful to populate this information appropriately when creating the **F RecType** records.*

OnDate (**Send Date:**) - stores the date on which the user is scheduled to perform this activity.

OnTime - as stated previously the **Literature FulFillment Requests** are untimed activities, as such this field should always remain blank for this **RecType**, unless one selects the ☐ **Alarm** option. For some unknown reason, this field is populated with 09:00 when the ☐ **Alarm** option is selected.

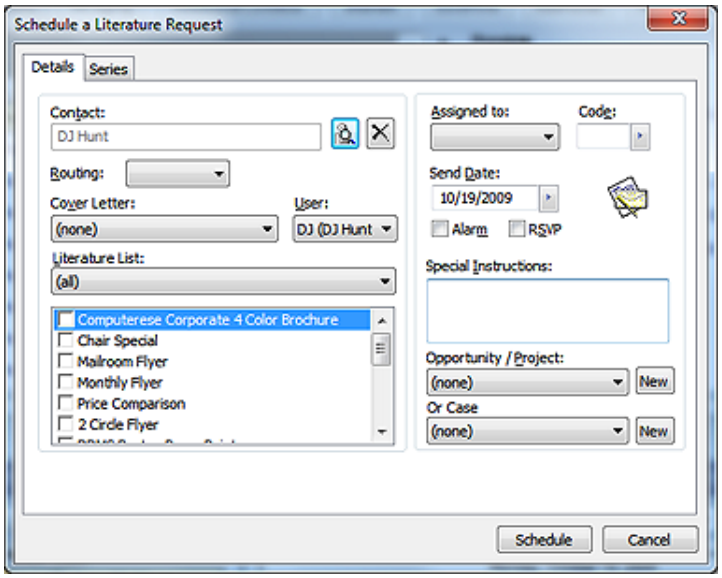


Figure 8-9

EndDate - is/should be populated with the **OnDate** value for this **RecType**.

AlarmFlag (☐ **Alarm**)- this field must always be populated with either an **N** for no alarm or a **Y** if the ☐ **Alarm** option has been selected.

AlarmTime - would contain the time an alarm is to be triggered, if any. Again, this time is entered into the table in a 24 hour format (i.e. **09:00**). There is no alarm time field available on the dialog form, hence, the **AlarmTime** is set to **9:00am** (i.e. **09:00**).

AlarmDate - contains the date on which an alarm is to be triggered, if any. Again, there is no alarm date field available on this dialog form, and the field is automatically set to the value contained in the **OnDate** field. Programmers populating this field through an external application should be consistent with this paradigm.

ActvCode (**Code:**) - could contain up to three characters which act as an activity code. This helps to granularize your GoldMine data for more focused reporting.

RSVP (☐ **RSVP**) - is a one character field that must contain an **N** or a **Y** if the ☐ **RSVP** option has been selected.

Duration (**Routing:**) - being a **Float** based field, is always defaulted to **0**, however, the value is controlled by the **Routing:** selection where a **0** = **Printer**, and a **1** = **FAX**.

RecType - for this section of the chapter is **F**.

AConfirm - despite what the structure table mentions, I have not found this to be used anywhere in my **Cal** table. This field always appears to contain a **.null.** value.

ApptUser - There is no color coding available for this **RecType**, hence, this field is populated empty for this **RecType**.

Status - the 2nd byte will be either a **0** or a **1** if there are no notes or if there are notes, respectively. The 2nd byte must always contain a value. For this rectype there are almost always **Notes**, as I will describe later, hence this value should always be a **1**.

DirCode - is, quite simply, the **File Code** for the contact dataset for which this activity has been scheduled against. In the past, this field was not employed. Today, the information in this field has greater significance, hence should be populated. The **DirCode** will be found in the **SpFiles** table or through the GUI:

Tools
Databases
Open Databases...

Number1 - for this **RecType** there is no option to mark the activity as **Private**. As this field is a **Float** based field the default value of **0** is displayed.

Number2 - for the task will always be **0**. As this is a **Float** based field, the field will always contain a **0**, and there is no need for an application developer to populate this field externally.

Company - this field is used to store the **Contact** name, or **Company** name when the **Contact** field is blank, of the contact record for whom this activity was scheduled against. As stated previously, this **RecType** must always be scheduled against a contact record.

Ref (**Special Instructions:**) - is up to 80 characters of a short description for this scheduled activity. This field is populated by the information contained in the **Special Instructions:** in the dialog form shown in Figure 8-9 on the previous page. Take special note, even though the dialog form makes this appear to be a text based field, it is not. Your users will be able to type continuously, however, when saved, only the first 80 characters of the **Special Instructions:** are retained.

Notes (**Literature List:**) - for this **RecType**, is used to store the requested literature, along with the **RecID** relating to that piece of literature as stored in the **Forms** table. A typical example of this might be:

```
Computerese Corporate 4 Color Brochure 8K72TH5)!B+!(+;
Chair Special 7A7OO9N(@M(][_V
Mailroom Flyer 9JIWR65*DDV2[_V
Monthly Flyer 7A7OMC9&9&3M[_V
Price Comparison 7A7OPC2&%JRG[_V
```

LinkRecID (**Cover Letter:**) - will maintain the **RecID** from the **Forms** table item, if the creator of the activity has selected to include a **Cover Letter:** in their **Literature FulFillment Request**.

LDocRecID - for this **RecType**, I have never seen this field populated with anything other than the default **.null.** value.

LOpRecID - employed when the activity is associated with an **Opportunity**/**Project**/**Case**, and, if there is such an association, this field must contain the **RecID** of the **Opportunity**/**Project** record or the **Case** record. As this is only one field a **Literature Request** can only be associated with either an **Opportunity**, a **Project** or a **Case**. One cannot associate an **Opportunity** and a **Case** for instance. An external application must populate this field with a space followed by a generated **RecID** if there is no association (i.e. **space(1)+9F6VPQ2%+IX]R#**).

CreateBy, **CreateOn**, **CreateAt**, **LastUser**, **LastDate**, **LastTime**, **Ext**, **Service**, and **RecID** - are all employee in the normal manner.

Note

*Programmers take note, each line in the Notes field for **RecType = F** is terminated with a **chr(10)**, while **chr(9)**, the tab character, is used to separate the document name from the **Form. RecID**.*

WARNING

*Programmers please note that the **LOpRecID** field must be populated. If there is no linked opportunity, then this field **must** have a space as the 1st character followed by an API generated **RecID**. This requirement was included to help with indexing, hence performance.*

Holiday

Cal.Rectype = H is used exclusively by GoldMine to maintain the various **Holiday** records that the administrator may choose to have displayed on the graphical calendar. There are only a few fields that are populated in the **Cal** table, and I will cover them here. However, you should keep in mind that the holidays are added to the **Cal** table via the **Option** GUI of a **UserID** possessing **Master Rights** within GoldMine.

UserID - maintains the GoldMine derived **UserID** for the holiday set. This takes the form of sequential, character based numbering, **000** through **012** in GoldMine Premium.

AccountNo - for this **RecType**, will always be **Holiday**.

RecType - for this **RecType**, will always be **H**.

Status - notice on this **RecType**, even though there are notes, that the 2nd character flag is not set as in the past **RecTypes** that we have discussed. On the other hand, notice that the 1st character flag is set to either a **2** or a **4**. FrontRange has stated that this is the flag value for categories:

2 - country (from the predefined set eg..US, UK etc)
4 - religion (Christian, Jewish etc)
8 - userdef category

Ref - contains the label assigned to the holiday set, and it is this label that is displayed when selecting the **Holiday** group via the **Options**.

Notes - this field contains the defined holidays associated with the given category. Here is an example of the information contained therein, and the astute observer will notice that there are beginning and ending tags, reminiscent of XML coding:

```
BEGIN:VCALENDAR
PRODID:-//FrontRange Solutions//GoldMine 8.0//EN
VERSION:2.0
METHOD:PUBLISH
   BEGIN:VEVENT
```

Note

*Indentation is for presentation only. In the **Notes** field, no such formatting is encountered.*

Note

*Indentation is for presentation only. In the **Notes** field, no such formatting is encountered.*

```
   ORGANIZER;CN="MASTER ():MAILTO:--X-GM-USER--<MASTER>
   DTSTART;VALUE=DATE:20020101
   DTEND;VALUE=DATE:20020102
   RRULE:FREQ=YEARLY;COUNT=30;INTERVAL=1;BYMONTHDAY=1;BYMONTH=1;WKST=SU
   TRANSP:OPAQUE
   SEQUENCE:0
   UID:39324D5057424A294A3C5D30303031
   DTSTAMP:20020722T081234Z
   SUMMARY:New Year's Day
   PRIORITY:5
   CLASS:PUBLIC
   CATEGORIES:HOLIDAY
END:VEVENT
BEGIN:VEVENT
   ORGANIZER;CN="MASTER ():MAILTO:--X-GM-USER--<MASTER>
   DTSTART;VALUE=DATE:20020121
   DTEND;VALUE=DATE:20020122
   RRULE:FREQ=YEARLY;COUNT=30;INTERVAL=1;BYDAY=MO;BYMONTH=1;BYSETPOS=3;
                WKST=SU
   TRANSP:OPAQUE
   SEQUENCE:0
   UID:39334C4B45394A284126285F462636
   DTSTAMP:20020722T081234Z
   SUMMARY:Martin Luther King Jr. Day
   PRIORITY:5
   CLASS:PUBLIC
   CATEGORIES:HOLIDAY
END:VEVENT
BEGIN:VEVENT
   ORGANIZER;CN="MASTER ():MAILTO:--X-GM-USER--<MASTER>
   DTSTART;VALUE=DATE:20020218
   DTEND;VALUE=DATE:20020219
   RRULE:FREQ=YEARLY;COUNT=30;INTERVAL=1;BYDAY=MO;BYMONTH=2;BYSETPOS=3;
                WKST=SU
   TRANSP:OPAQUE
   SEQUENCE:0
   UID:39334C4B48343326602A3236462636
   DTSTAMP:20020722T081234Z
   SUMMARY:Presidents Day
   PRIORITY:5
   CLASS:PUBLIC
   CATEGORIES:HOLIDAY
END:VEVENT
BEGIN:VEVENT
   ORGANIZER;CN="MASTER ():MAILTO:--X-GM-USER--<MASTER>
   DTSTART;VALUE=DATE:20020527
   DTEND;VALUE=DATE:20020528
   RRULE:FREQ=YEARLY;COUNT=30;INTERVAL=1;BYDAY=MO;BYMONTH=5;BYSETPOS=-1;
               WKST=SU
   TRANSP:OPAQUE
   SEQUENCE:0
   UID:39334C4B484D502441253235462636
   DTSTAMP:20020722T081234Z
   SUMMARY:Memorial Day
   PRIORITY:5
   CLASS:PUBLIC
   CATEGORIES:HOLIDAY
END:VEVENT
BEGIN:VEVENT
   ORGANIZER;CN="MASTER ():MAILTO:--X-GM-USER--<MASTER>
   DTSTART;VALUE=DATE:20020704
   DTEND;VALUE=DATE:20020705
   RRULE:FREQ=YEARLY;COUNT=30;INTERVAL=1;BYMONTHDAY=4;BYMONTH=7;WKST=SU
   TRANSP:OPAQUE
   SEQUENCE:0
   UID:39334C4B4941302A3B343635462636
   DTSTAMP:20020722T081234Z
   SUMMARY:Independence Day
   PRIORITY:5
   CLASS:PUBLIC
   CATEGORIES:HOLIDAY
END:VEVENT
BEGIN:VEVENT
   ORGANIZER;CN="MASTER ():MAILTO:--X-GM-USER--<MASTER>
   DTSTART;VALUE=DATE:20020902
   DTEND;VALUE=DATE:20020903
   RRULE:FREQ=YEARLY;COUNT=30;INTERVAL=1;BYDAY=MO;BYMONTH=9;BYSETPOS=1;
                WKST=SU
   TRANSP:OPAQUE
```

Note

*Indentation is for presentation only. In the **Notes** field, no such formatting is encountered.*

```
    SEQUENCE:0
    UID:39334C4B4959462547384l4A462636
    DTSTAMP:20020722T081234Z
    SUMMARY:Labor Day
    PRIORITY:5
    CLASS:PUBLIC
    CATEGORIES:HOLIDAY
  END:VEVENT
  BEGIN:VEVENT
    ORGANIZER;CN="MASTER ():MAILTO:--X-GM-USER--<MASTER>
    DTSTART;VALUE=DATE:20021014
    DTEND;VALUE=DATE:20021015
    RRULE:FREQ=YEARLY;COUNT=30;INTERVAL=1;BYDAY=MO;BYMONTH=10;BYSETPOS=2;
          WKST=SU
    TRANSP:OPAQUE
    SEQUENCE:0
    UID:39334C4B4C4433245C4A3448462636
    DTSTAMP:20020722T081234Z
    SUMMARY:Columbus Day
    PRIORITY:5
    CLASS:PUBLIC
    CATEGORIES:HOLIDAY
  END:VEVENT
  BEGIN:VEVENT
    ORGANIZER;CN="MASTER ():MAILTO:--X-GM-USER--<MASTER>
    DTSTART;VALUE=DATE:20021111
    DTEND;VALUE=DATE:20021112
    RRULE:FREQ=YEARLY;COUNT=30;INTERVAL=1;BYMONTHDAY=11;BYMONTH=11;WKST=SU
    TRANSP:OPAQUE
    SEQUENCE:0
    UID:39334C4B4E305A243B327B2A462636
    DTSTAMP:20020722T081234Z
    SUMMARY:Veterans Day
    PRIORITY:5
    CLASS:PUBLIC
    CATEGORIES:HOLIDAY
  END:VEVENT
  BEGIN:VEVENT
    ORGANIZER;CN="MASTER ():MAILTO:--X-GM-USER--<MASTER>
    DTSTART;VALUE=DATE:20021128
    DTEND;VALUE=DATE:20021129
    RRULE:FREQ=YEARLY;COUNT=30;INTERVAL=1;BYDAY=TH;BYMONTH=11;BYSETPOS=4;
          WKST=SU
    TRANSP:OPAQUE
    SEQUENCE:0
    UID:39334C4B4F3446244F234056462636
    DTSTAMP:20020722T081234Z
    SUMMARY:Thanksgiving Day
    PRIORITY:5
    CLASS:PUBLIC
    CATEGORIES:HOLIDAY
  END:VEVENT
END:VCALENDAR
```

You may have noticed that there are **9** predefined holidays in this text file, and they happen to be in the **Cal.Notes** field for the **Cal.Ref** = **United States** record.

WARNING

*Programmers, please note that the **LOpRecID** field must be populated. If there is no linked opportunity, then this field **must** have a space as the 1st character followed by an API generated **RecID**. This requirement was included to help with indexing, hence performance.*

LOpRecID - employed when the activity is associated with an **Opportunity**/**Project**/**Case**, and, if there is such an association, this field must contain the **RecID** of the **Opportunity**/**Project** record or the **Case** record. As the **Holiday RecType** has no link, an external application must populate this field with a space followed by a generated **RecID** (i.e. **space(1)+9F6VPQ2%+IX]R#**).

CreateBy, **CreateOn**, **CreateAt**, **LastUser**, **LastDate**, **LastTime**, **Ext**, **Service**, and **RecID** - are all employee in the normal manner.

E-mail, Queued E-mail Messages & Quotas

Note

*GoldMine will use **RecType** = [**Q**] for both **Queued E-mails** waiting to be delivered, and **Quotas** for a **UserID**.*

Cal.Rectype = **M** and **Q** have a lot in common as **M** is for E-mail messages while **Q** is for Queued E-mail messages. Structurally, these two record types are virtually identical. One thing that an Administrator/Developer should understand is that these record types in the **Cal** table, and the **ContHist.sRecType** of **M** in the **ContHist** table, are little more then related referential pointers to the actual messages that reside in the **MailBox** table. These are **not** the actual messages themselves that are being sent/received.

UserID - maintains the GoldMine **UserID** for the user associated with the **From: /M** or **To: /Q** account in the e-mail itself. When **Q** is used for **Quota**, this field is blank.

AccountNo - stores the relational link back to the appropriate record in the **Contact1** table. Outgoing messages are ☑ **Link**ed by default, and incoming messages are linked to a record if the

e-mail address is contained within the GoldMine dataset, and the user Options has been selected to ☑ **Link e-mail address to contact record by default** on the **E-Mail** | **Advanced** tab. Having said this, then you can see that it is possible for this field to be **.null.**, as would be the case with unlinked outgoing/incoming e-mails. When **Q** is used for **Quota**, this field looks something like this:

QSDJ 20070731

Where the **QS** is followed by the full 8 characters of the userid, and then the **To Date:** for the quota period.

OnDate - stores the date from the **Date** value contained in the header section of the e-mail. When **Q** is used for **Quota**, this field contains the **From Date:** for the quota period.

OnTime - stores the time portion of the **Date** value contained in the header section of the e-mail. When **Q** is used for **Quota**, this field is used for the first 5 characters of the userid.

EndDate - is/should be populated with the **Cal.OnDate** value for this **RecType**.

AlarmFlag - this field must always be populated with either an **N** for no alarm or a **Y** if there is an alarm. This field can be set for outgoing Internet e-mail, but is irrelevant, as it is not received by the e-mail recipient. As the dialog form is used for both GoldMine Internal E-mail and Internet E-mail the ☐ **Alarm** checkbox is really only applicable to the GoldMine Internal E-mail. In some cases, i.e. GoldMine sending the user **RSVP**, you may not see the normal **N** value. I have noticed, on occasion, that the GoldMine application, itself, is inserting an ***** into this field.

AlarmTime - contains the time an alarm is to be triggered, if any. Again, this time is entered into the table in a 24 hour format (**i.e. 14:00**). There is no alarm time field available on the dialog form, hence, the **Cal.AlarmTime** is set to the send time minus the user preference decrement, default 10 minutes (**i.e. 13:50**). Programmers populating this field through an external application should be consistent with this paradigm, and refer to **Cal.AlarmFlag**.

AlarmDate - contains the date on which an alarm is to be triggered, if any. Again, there is no alarm date field available on this dialog form, and the field is automatically set to the value contained in the **OnDate** field. Programmers populating this field through an external application should be consistent with this paradigm, and refer to **AlarmFlag**.

ActvCode - there is no place to enter a **Code** unless Queuing a message where the field is called **Activity**. Retrieved Internet E-mail will not have this field populated.

RSVP - is a one character field that must contain an **N** or a **Y** if this option has been selected, however, as with **Cal.AlarmFlag** this field is only relevant for GoldMine Internal E-mail, though it will show up in queued Internet e-mail. This field can be set for outgoing Internet e-mail, but is irrelevant as it is not received by the e-mail recipient. As the dialog form is used for both GoldMine Internal E-mail and Internet E-mail the ☐ **RSVP** checkbox is really applicable to GoldMine Internal E-mail only. In some cases, i.e. GoldMine sending the user **RSVP**, you may not see the normal **N** value.

Duration - being a Float based field, is always defaulted to **0**, and is not used with this **Cal.RecType**.

RecType - for this section of the chapter could be an **M** or **Q**.

AConfirm - despite what the structure table mentions, I have not found this to be used anywhere in our **Cal** table.

ApptUser - There is no color coding available for these rectypes, hence, this field is not populated for these rectypes with exceptions. When GoldMine, itself, is sending an **RSVP** to an activity which did posses a color coding the **Cal.ApptUser** field will also mimic that color coding.

Status - the second byte will be a **0**. The 2nd byte must always contain a value. For these rectypes, there are no notes as this record is only a pointer to the **Mailbox** e-mail message record.

DirCode - is, quite simply, the **File Code** for the contact dataset for which this activity has been scheduled against. In the past, this field was not employed. Today, the information in this field has significance and, hence, should be populated. The **DirCode** will be found in the **SpFiles** table.

Number1 - for these rectypes defaults to **7**. If **Private**, this value is the sum of **7** + **16** or **23**. I have determined, with assistance from Paul Redstone, that the setting of the queue options of an Outgoing Internet E-mail determines the base value of this field.

Here is the base number as defined by the option selection.

7 = ☑ Create a history record	☑ Save the body text
6 = ☐ Create a history record	☑ Save the body text
5 = ☑ Create a history record	☐ Save the body text
4 = ☐ Create a history record	☐ Save the body text

Number2 - for the task will always be **0**. As this is a Float based field, the field will always contain a **0**, and there is no need for an application developer to populate this field externally.

Company - unlike the **Company** field for other **Cal.RecType** values, the **Company** field for these rectypes comes from the **Contact's name:** in the **ContSupp** table that is associated with the **From:** address of the e-mail. When **Q** is used for **Quota**, this field is used for storing the **Quota:**, **Forecast:**, **Closed Sales:** and **Lost Sales:** values. There are 60 characters in this field, and each value could be up to 15 characters in length with no special characters being used.

Ref - up to 80 characters of the **Subject:** line from the e-mail message retrieved or queued.

Notes - There are no **Notes** for these rectypes. Remember that the e-mail message is stored in total in **Mailbox.RFC822**, and that there is only a link to that **Mailbox** record contained in the **Cal** record for these rectypes.

LinkRecID - will maintain the **Mialbox.RecID** for the associated/linked **Mailbox** record for these rectypes.

LDocRecID - for this rectype, we have never seen this field populated, and it appears to maintain the default **.null.** value.

LOpRecID - as there can now be an association with an **Opportunity**/**Project**/**Case**, then this field could be populated with the related **RecID** for the **Opportunity**/**Project**/**Case**. Any external application populating this rectype record must populate this field with a space followed by a generated **RecID**. (**i.e. space(1)+9F6VPQ2%+IX]R#**)

CreateBy, **CreateOn**, **CreateAt**, **LastUser**, **LastDate**, **LastTime**, **EXT**, **SERVICE**, and **RecID** - are all employed in the normal manner. **CreateBy** for received Internet e-mails is always **INTERNET**.

Forecasted Sales

Cal.RecType = **S** is the last rectype that I will be covering with respect to the **Cal** table. This rectype stands for, appropriately enough, **Sale** as the GoldMine menu displays it.

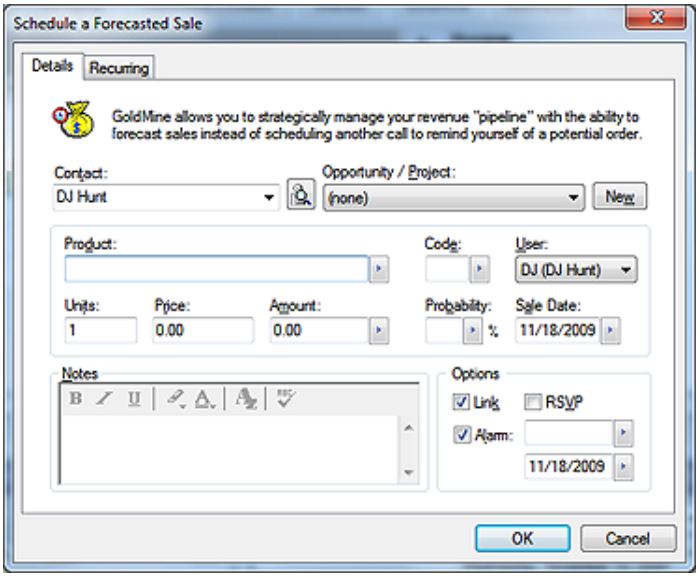


Figure 8-10

UserID (**User:**) - maintains the GoldMine **UserID** for the user associated with the sale.

AccountNo - stores the relational link back to the appropriate record in the **Contact1** table. Scheduling a **Sale** is ☑ **Linked** by default although it is possible to have an unlinked sale (no **AccountNo**) it is certainly never recommended as this tends to defeat the premise behind a CRM solution.

OnDate (**Sale Date:**) - stores the **Sale Date:** which is better known as the expected close date for the sale. The key words here are "expected close date".

OnTime - Forecasted Sales are untimed activities that display under the Task list in the graphical calendar, hence, this field is usually empty (not **.null.**). However, this does not prevent the programmer from placing a value into this field causing their Forecasted Sale to be rendered in the timed activity area of the graphical calendar. Again, this time would be entered into the table in a 24 hour format (**i.e. 14:00**).

EndDate - is and should be populated with the **Cal.OnDate** value for this rectype.

AlarmFlag - this field must always be populated with either an **N** for no alarm or a **Y** if the ☐ **Alarm:** option has been selected.

AlarmTime - contains the time an alarm is to be triggered, if any. Again, this time is entered into the table in a 24 hour format (**i.e. 14:00**).

AlarmDate - contains the date on which an alarm is to be triggered, if any.

ActvCode (**Code:**) - could contain up to three characters which act as an activity code for the given activity. This helps to granularize your GoldMine data for more focused reporting.

RSVP (☐ **RSVP**) - is a one character field that must contain an **N** if the ☐ **RSVP:** option has not been selected in the scheduling dialog form, and a **Y** if this option has been selected.

Duration (**Probability:**) - being a Float based field, is always defaulted to **0**, however, for this rectype this field maintains the **Probability:** value as entered via the dialog form, and is used in many reports within GoldMine.

RecType - for this section of the chapter must be **S**

AConfirm - despite what the structure table mentions, we have not found this to be used anywhere in our **Cal** table.

ApptUser - There is no color coding available for this rectype as this type of activity is displayed in the untimed Task area of the graphical calendar, hence, this field is not populated for this rectype. However, the programmers amoung us, that have placed this activity in the timed area of the graphical calendar, may wish to color code this activity (**Green = $**).

Status - the 1st byte in this field may contain a **B** if this activity is one of multiple activities that were scheduled at the same time, while the second byte will be either a **0** or a **1** if there are no notes, or if there are notes, respectively. The 2nd byte must always contain a value regardless.

DirCode - is, quite simply, the **File Code** for the contact dataset for which this activity has been scheduled against. In the past, this field was not employed. Today, the information in this field has significance and, hence, should be populated. The **DirCode** will be found in the **SpFiles** table.

Number1 (**Amount:**) - for this rectype, this field is used to store the **Amount:** as supplied from the dialog form shown in Figure 8-10 on the previous page. One must remember that this is the value contained in the **Units:**, refer to **Cal.Number2**, times the value of the **Price:** as displayed in the same dialog form. As this field is being used to store the amount of the sale, there is no option for privatizing this type of rectype.

Number2 (**Units:**) - for this rectype, this field stores the **Units:** as supplied from the dialog form.

Company (**Contact:**) - this field is used to store the **Contact** name, or **Company** name when the **Contact** field is blank, of the contact record for whom the activity was scheduled against.

Ref (**Product:**) - is up to 80 characters of a short description for this scheduled activity.

Notes (**Notes:**) - is employed to hold any plain text or rich text type **Notes:** that were entered for this activity. Be mindful, in GoldMine Premium that these notes will be in an HTML format unless that option was specifically turned off in the **GM.ini** as I discussed in Chapter 3, although it is now preferrable to use rich text (HTML) Notes.

LinkRecID - will remain empty for this rectype.

LDocRecID - for this rectype we have never seen this field populated, and retains the default **.null.** value.

LOpRecID - as there could be an association with an Opportunity/Project, this field could contain the RecID for the related Opportunity/Project. Otherwise, the default value is a space followed by a generated RecID. (**i.e. space(1)+9F6VPQ2%+IX]R#**)

CreateBy, **CreateOn**, **CreateAt**, **LastUser**, **LastDate**, **LastTime**, **EXT**, **SERVICE**, and **RecID** - are all employed in the normal manner.

WARNING

*Programmers please note that the **LOpRecID** field must be populated. If there is no linked opportunity, then this field **must** have a space as the 1st character followed by an API generated **RecID**. This requirement was included to help with indexing, hence performance.*

MailBox

MailBox Table

Indexes

MBOXACCNO	**AccountNo** (Non-Unique, Non-Clustered)
MBOXFOLDER	**Folder** (Non-Unique, Non-Clustered)
MBOXLINK	**LinkRecID** (Non-Unique, Non-Clustered)
MBOXMAILID	**MailID** (Non-Unique, Non-Clustered)
MBOXUSER	**UserID, Folder, Folder2, MailDate** (Non-Unique, Non-Clustered)
MBXRECID	**RecID - Unique** (Unique, Non-Clustered)

Relationships

MailBox.RecID	Many-to-One	**Cal.LinkRedID**
MailBox.RecID	Many-to-One	**ContHist.LinkRecID**
MailBox.AccountNo	Many-to-One	**Contact1.AccountNo**
MailBox.LOpRecID	One-to-One	**OpMgr.RecID**
MailBox.LOpRecID	One-to-One	**Cases.RecID**

Structure

AccountNo	**VarChar**	**20**	**Linked Contact AccountNo -** not Null
CreateOn	**DateTime**	**8**	**Creation Date**
Ext	**VarChar**	**5**	**Type**
Flags	**VarChar**	**8**	**Flags***
Folder	**VarChar**	**20**	**Folder identification** ** - not Null
Folder2	**VarChar**	**20**	**Subfolder identification -** not Null
LinkRecID	**VarChar**	**15**	**RecID of linked Cal\ContHist record -** not Null
LOpRecID	**VarChar**	**15**	**RecID of linked OpMgr/Cases record**
MailDate	**DateTime**	**8**	**Mail Date**
MailID	**VarChar**	**200**	**Mail indentification -** not Null
MailSize	**VarChar**	**8**	**Mail Size**
MailTime	**VarChar**	**8**	**Mail Time**
MailRef	**VarChar**	**100**	**Reference**
RFC822	**Image**	**16**	**Entire mail message including coding**
RecID	**VarChar**	**15**	**Record ID -** not Null
UserID	**VarChar**	**8**	**GoldMine UserID -** not Null

Notes

*** The flags field, although character based, contains the the converted binary number which relates to the following states:**

Bit	On	Off
1	Read	Not Read
2	In History	Not in History
3	Outbound	Inbound
4	Attachments	No Attachments

**** The Folder field contains the name of the folder in which mail is stored. There are several predefined folder names:**

X-GM-FOLDERS	-	Main Level Folder
X-GM-GROUPS	-	Groups (Legacy Identifier)
X-GM-HTMLTAB	-	GM+Views
X-GM-ICALINFO	-	iCalendar Information
X-GM-INBOX	-	Inbox
X-GM-PROP-HTMLTAB	-	HTML Tab Settings
X-GM-OUTBOX	-	Outbox
X-GM-RULES	-	E-mail Rules
X-GM-SUBXXXXX	-	Subfolder of a Main Level Folder, XXXXX might be FILED
X-GM-SURX-GMRXXXXXXX	-	Subsubfolder, XXXXXXX = left 7 Chr of the Subfolder RecID
X-GM-TD-ITEMS	-	MyGoldMine Items
X-GM-TD-SETTINGS	-	MyGoldMine Settings
X-GM-TEMPLATES	-	E-mail Templates
X-GM-TRASH	-	Trash

AccountNo - stores the relational link back to the appropriate record in the **Contact1** table. This field may be blank in the **MailBox** table if the message was not linked. Additionally this field may contain the display name for a GoldMine predefined Folder.

Folder	ACCOUNTNO
X-GM-FOLDERS	**Main Level folder name**
X-GM-SUBXXXXX	**Main Level folder name for this subfolder**
X-GM-SURX-GMRXXXXXXX	**X-GMRXXXXXXXXXXXXXXX RecID for folder**

CreateOn - as in all other tables is the creation date of the record. Depending on the Users Options this may be the date that this record was created in GoldMine or it may be the date from the e-mails header.

Flags - which is actually a character based field, and holds a numeric value. This value is derived from the binary number that represents the value for the four states (see table on previous page).

Folder - may contain any of the default GoldMine folder identifiers (see table on the previous page). This field may also contain any user defined top level folder names. Said folders may contain sub-folders.

Folder2 - contents is related to **Folder** contents, and may contain:

Folder	Folder2
X-GM-FOLDERS	Empty
X-GM-GROUPS	Empty
X-GM-HTMLTAB	GM+Views template name
X-GM-ICALINFO	iCal request information
X-GM-INBOX	Empty
X-GM-OUTBOX	Empty
X-GM-PROP-HTMLTAB	Empty
X-GM-RULES	Empty
X-GM-SUBXXXXX	Subfolder name
X-GM-SURX-GMRXXXXXXX	Subsubfolder name
X-GM-TD-ITEMS	Empty
X-GM-TD-SETTING	Empty
X-GM-TEMPLATES	Empty
X-GM-TRASH	Empty

Note

***Solica Consulting Limited**, at **http://www.Solica.com** has developed a tool to fix the "**Cal record not found**" errors. This tool, **FixCal**, is distributed freely, and you use it at your own risk.*

Note

*Unlike other tables, the **Mailbox.LOpRecID** field does not need to be populated when there is no relationship to an **Opportunity**/**Project**/**Case**.*

LinkRecID - whenever there is an e-mail in the **Mailbox** table there is also a related record in the **Cal** table if the e-mail has not been **Filed**, and if it has been filed, then the relationship is to the **ContHist** table. The **RecID** of the record from the related table is stored in this field, and, correspondingly, the **RecID** from the e-mail record, in the **MailBox** table, is stored in the **LinkRecID** field of the related record. Occasionally, you may encounter a "**Cal record not found.**" error. There is no known cause as to why the link breaks, only that it does get broken occasionally (refer to sidebar).

LOpRecID - maintains any links that may result with this e-mail being associated with an Opportunity/Project/Case. This field would contain the RecID of the linked Opportunity/Project/Case.

MailDate - holds the date which is recorded for the e-mail in its header if the user has selected ☑ **Use date from mail header** in their GoldMne Options, otherwise this field should contain the retrieval date for any given message.

MailID - This field was newly added in 8.5.1.12, and contains the **E-mail Identification** for a given e-mail message. This should prevent GoldMine from downloading an e-mail message that it had previously downloaded (no more duplcates hoefully). Syntactically the ID could resemble:

```
<RDQ1UIVBVSF7QD85UiNZMzA3OTgxOA@DJ>
```

MailSize - retains the size in bytes of the e-mail message.

MailTime - holds the time which is recorded for the e-mail in its header if the user has selected ☑ **Use date from mail header** in their GoldMne Options, otherwise this field should contain the retrieval time for any given message.

MailRef - maintains a concatenated string which is the sum of the **From:** value in the e-mail header, the **Tab** character, **chr(9)**, and the **Subject:** value in the e-mail header.

RFC822 - a memo type field that maintains the whole enchilada, so to speak. I believe that **RFC822** was an e-mail format type, hence the name of the field. This field contains the e-mail in its entirety. Here is a sample message:

```
Date: Wed, 18 Nov 2009 19:24:15 -0500
From: "Kelly & Danny" <k41875@djhunt.us>
Subject: Eric Superbowl Game
To: <DJ@DJHunt.US>
Message-ID: <9E2E0FC0CFF04FE6A342B5967AAC949D@homebb7fb904c2>
Mime-Version: 1.0  X-Mailer: Microsoft Office Outlook 11
Received: (qmail 16595 invoked by uid 78); 19 Nov 2009 00:26:52 -0000
Received: from unknown (HELO cloudmark1) (10.49.16.78)  by 0 with SMTP; 19 Nov 2009 00:26:52
Return-Path: <k41875@djhunt.us>
Received: from [206.46.173.1] ([206.46.173.1:53871] helo=vms173001.mailsrvcs.net) by cm-mr4
(envelope-from <k41875@djhunt.us>) (ecelerity 2.2.2.41 r(31179/31189)) with ESMTP id E7/11-00355-
CC0940B4; Wed, 18 Nov 2009 19:26:52 -0500
Received: from homebb7fb904c2 ([unknown] [72.85.204.48]) by vms173001.mailsrvcs.net (Sun
Java(tm) System Messaging Server 7u2-7.02 32bit (built Apr 16 2009)) with ESMTPA id <0KTB-
00K6UYGGXCC1@vms173001.mailsrvcs.net> for DJ@DJHunt.US; Wed, 18 Nov 2009 18:26:51-0600
```

Thread-index: AcporqBZwlEkhZcjTBi4lxfeTtAfxw==
X-MIMEOLE: Produced By Microsoft MimeOLE V6.00.2900.5579
Content-Type: multipart/alternative; boundary="----=_NextPart_000_00FA_01CA6884.BBC48230"
Content-Transfer-Encoding: 8bit
------=_NextPart_000_00FA_01CA6884.BBC48230
Content-Type: text/plain; charset="us-ascii"
Content-Transfer-Encoding: 8bit Sunday at 8:30 am At 10 Elmwood Place Boylston, Ma
There will be an admission fee, last game of the season We are the home team and park on the west side
------=_NextPart_000_00FA_01CA6884.BBC48230
Content-Type: text/html; charset="us-ascii"
Content-Transfer-Encoding: 8bi
<html xmlns:o="urn:schemas-microsoft-com:office:office" xmlns:w="urn:schemas-microsoft-com:office:word" xmlns:st1="urn:schemas-microsoft-com:office:smarttags" xmlns="http://www.w3.org/TR/REC-html40">
<head>
<META HTTP-EQUIV="Content-Type" CONTENT="text/html; charset=us-ascii">
<meta name=Generator content="Microsoft Word 11 (filtered medium)">
<o:SmartTagType namespaceuri="urn:schemas-microsoft-com:office:smarttags" name="Street"/>
<o:SmartTagType namespaceuri="urn:schemas-microsoft-com:office:smarttags" name="address"/>
<!--[if !mso]> <style> st1\:*{behavior:url(#default#ieooui) } </style> <![endif]--> <style>
<!-- /* Style Definitions */ p.MsoNormal, li.MsoNormal, div.MsoNormal {margin:0in; margin-bottom:.0001pt; font-size:12.0pt; font-family:"Times New Roman";} a:link, span.MsoHyperlink {color:blue; text-decoration:underline;} a:visited, span.MsoHyperlinkFollowed {color:purple; text-decoration:underline;} span.EmailStyle17 {mso-style-type:personal-compose; font-family:Arial; color:windowtext;} @page Section1 {size:8.5in 11.0in; margin:.5in .5in .5in .5in;} div.Section1 {page:Section1;} --> </style>
</head>
<body lang=EN-US link=blue vlink=purple>

Sunday at 8:30 am<o:p></o:p>
<o:p> </o:p>
At <st1:Street w:st="on"><st1:address w:st="on">10 Elmwood Place</st1:address></st1:Street><o:p></o:p>
Boylston, Ma<o:p></o:p>
<o:p> </o:p>
There will be an admission fee, last game of the season<o:p></o:p>
<o:p> </o:p>
We are the home team and park on the west side<o:p></o:p>
</body>
</html>
------=_NextPart_000_00FA_01CA6884.BBC48230--

RecID - at this point in the chapter, I probably don't need to explain that this is a unique record identification that is generated by GoldMine for each and every record in each and every table within GoldMine. Programmers, using the API, will find that this value is generated automatically when they Append a record through the API.

UserID - maintains the GoldMine user identification for the user who was the creator of this message record in the **Mailbox** via whatever means.

Cases

Note

*There is a significant structural change as of at least build 8.5.2.5, whereby many fields that were previously **not Null** are today capable of accepting **Null** values.*

CASES Table

Indexes

CASEACC	**AccountNo** (Non-Unique, Non-Clustered)
CASENUM	**Number** (Non-Unique, Non-Clustered)
CASEOWNER	**Owner** (Non-Unique, Non-Clustered)
CASERECID	**RecID** (Unique, Non-Clustered)

Relationships

Cases.RecID	One-to-Many	**CaseTeamLink.CaseID**
Cases.RecID	One-to-Many	**CaseInfoLink.CaseID**
Cases.RecID	One-to-Many	**CaseAttachment.CaseID**
Cases.RecID	Many-to-One	**Cal.LOpRecID**
Cases.AccountNo	One-to-One	**Contact1.AccountNo**

Structure

AccountNo	**VarChar**	**20**	**Linked Contact AccountNo** - not Null
Category	**VarChar**	**40**	**Category of Case**
Created_By	**VarChar**	**8**	**UserID**
Created_Date	**DateTime**	**8**	**Case Creation Date**
Description	**Text**	**16**	**Description**
Due_Date	**DateTime**	**8**	**Autogenerated Due Date**
FType	**VarChar**	**40**	**Type of Case**
Is_Read	**SmallInt**	**2**	**Read** - not Null
Is_Template	**SmallInt**	**2**	**Template Identification** - not Null
Modified_By	**VarChar**	**8**	**Last Modified By**
Modified_Date	**DateTime**	**8**	**Last Modified Date**
Notes	**Text**	**16**	**Notes**
Number	**VarChar**	**40**	**Autogenerated Case Number** - not Null
Offering	**VarChar**	**200**	**Product/Service Associated with Case**
Owner	**VarChar**	**8**	**GoldMine UserID of Case Owner** - not Null
Priority	**VarChar**	**40**	**Priority of Case**
RecID	**VarChar**	**15**	**Record ID** - not Null
Resolution_Note	**Text**	**16**	**Resolution Note**
Resolution_Type	**VarChar**	**40**	**Resolution Type**
Resolved_By	**VarChar**	**8**	**UserID of Resolver**
Resolved_Date	**DateTime**	**8**	**Time Resolution Achieved**
Source	**VarChar**	**40**	**Source of Case**
Status	**SmallInt**	**2**	**Current State***
Subject	**VarChar**	**200**	**Subject of Case**

* The following are possible values for the Status field:

0 Template	**1** Assigned	**2** Reassigned
3 Escalated	**4** Resolved	**5** Abandoned

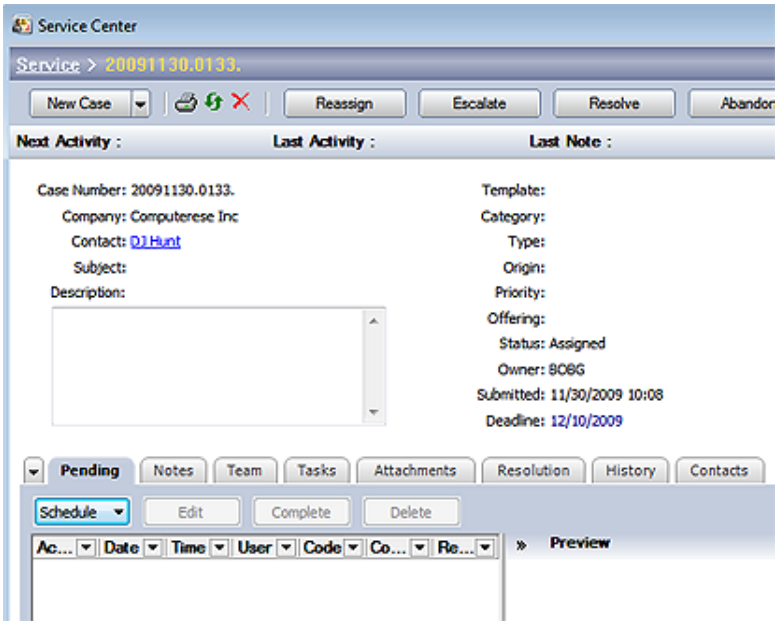


Figure 8-11

AccountNo - stores the relational link back to the appropriate record in the **Contact1** table. This field may be blank in the **Cases** table as in the **Case Template** record.

Note

*If your organization is not using **Case Templates** then I highly recommend that you populate your **F2 Lookup List** for each field so as to derive some form of consistency of data input.*

As we all know, this will assist you when reporting against this data.

Category (**Category:**) - stores up to 40 characters for your **Case Category**. This value may be predefined via your **Case Templates**. This helps to granularize your Cases read/opened by the assigned **Owner**.

Due_Date - self descriptive, and autogenerated based on the **Case Manager Preferences** dialog form **Default Follow-up Interval:** value.

FType - record type field that appears to be used when creating **Case Templates** when completing the **Case Type:** field.

Is_Read - self descriptive, however, this field stores a **0** or a **1**. If programmatically being set, the coder must make certain that one initiates this value to **0**.

Is_Template - self descriptive, however, this field stores a **0** if this is not a **Case Template** record or a **1** if this is a **Case Template** record. If programmatically being set, the coder must make certain that one initiates this value to **0**.

Notes (**Notes** tab) - being a Text based field, this field can store an almost infinite amount of information, and, although displayed differently then we have been used to in the past, it still contains the same information as we have all come to know and expect with the addition of a little HTML coding.

```
*** DJ (DJ Hunt) *** 12/2/2009 10:04:56 AM
<div>
      This ia a Note added under the Case Notes tab.
</div>
```

Number (**Case Number:**) - Each organization may configure this differently. A programmer will need to look at the **Cases.Number** field of the last **Cases** record created to see the current formatting configuration. You may expect that the second segment is the number that is incremented for each new Case.

Example: **DJ.0104.04112007**

Offering (**Offering(s):** Main Dialog | **Product/Service:** Template Dialog) - stores up to 200 characters for this fields value. Again, this value can be predefined in your **Case Templates**, and has an F2 Lookup list associated with the field as well. However, you are reminded that the F2 Lookup list values may not, themselves, exceed 40 characters which is a limitation of the **Lookup** table structure.

Owner (**Owner:**) - holds the **UserID** of the individual who is responsible for the flow of this particular Case record.

Priority (**Priority:**) - stores up to 40 characters for this fields value. Again, this value can be predefined in your **Case Templates**, and has an F2 Lookup list associated with the field as well.

Resolution_Note (**Resolution** tab) - being a Text based field, this field can store an almost infinate amount of information, and, although displayed differently than we have been used to in the past, it still contains the same information as we have all come to know and expect with the addition of a little HTML coding.

```
*** DJ (DJ Hunt) *** 12/2/2009 9:55:54 AM
<div>
   <FONT face=Tahoma size=2>
      <DIV>
         This is a test of a Resolved Case
      </DIV>
   </FONT>
</div>
```

Resolution_Type - currently I do not see that this field is being utilized, however, it is being defaulted to **empty** as opposed to having a **.null.** value.

Source (**Origin:**) - this value may contain up to 40 characters. Again, this value can be predefined in your **Case Templates**, and has an F2 Lookup list associated with the field as well if you are not using templates.

Status - refer to the footnote for the table on the previous page for the possible value that may appear in this field.

Subject (**Subject:**) - stores up to 200 characters for this fields value. Again, this value can be predefined in your **Case Templates**, and has an F2 Lookup list associated with the field as well. However, you are reminded that the F2 Lookup list values may not, themselves, exceed 40 characters.

Modified_By, **Modified_Date**, **Created_By**, **Created_Date**, **LastDate**, **Resolved_By**, **Resolved_Date**, and **RecID** - are all employed in the expected manner.

9

In This Chapter

Note

If you have purchased a paperback copy of this book, please forward a copy of your paid Invoice to:

DJ@DJHunt.US

Once I have proof of purchase, you will be sent the eBook version along with all of the documents at no extra charge.

Remember that the eBook version is Searchable, whereas the printed version can only be searched page by page.

WebImporting is a feature that has been included in the GoldMine application for some time, albeit, unsupported by FrontRange Solutions. In its first incarnation, WebImporting only allowed the users to add new contacts to their GoldMine from a web site form. One could also have the users update their GoldMine record in your database via an e-mail message (an example that I will discuss in this chapter). Computerese Incorporated also uses the WebImport script to generate e-mail messages that notify us that a user wishes to schedule a meeting. We also notify product vendors when a visitor has viewed their product on our web site, and requests additional information. The possibilities for WebImport are endless, and, again, you are only limited by your imagination. Alas, not many utilize this feature capability within GoldMine. If you maintain a web site, then there is no excuse for not capturing any registration leads from that web site directly into your GoldMine.

FrontRange supplied us with a sample web form, and the accompanying PERL script. We could modify the form to fit in with the theme of our web site, and place the PERL script on our web site for processing, assuming that your web site was capable of processing PERL (Linux Servers). Unfortunately, I was using a Windows Server so, as you'll see, my scripts needed to be written in ASP. The important thing to understand is the functionality of the script, and then you can script the code in any language with which you are familiar.

FrontRange has incorporated into GoldMine Premium the GoldMine WebImport - Script Generator. I have also supplied my own files to accompany my descriptions/definitions in this chapter. When you downloaded this book, the WebImportFiles.zip were included as part of the eBook download. If you've lost them, or you ordered the paperback version of this book, you may E-mail me directly and request the book inclusions. The file contains the documents that I will be discussing in detail in this chapter. **The Contact_Info.htm** document, and the **Process.asp** document to be specific.

Maybe I should explain the functional attributes of these documents before I actually get into them, and analyze them in some detail. I never, ever use the GoldMine HTML Editor for my E-mail Templates, or the GM+Views HTML Editor for that matter, as it does not maintain a legible structured format. Once created in GoldMine, I never know where to go in, and where to make any changes that I require. I, therefore, create all of my HTML documents in DreamWeaver, and then use the GoldMine **Include a text file** feature to include a pointer in our HTML document in the body of the GoldMine E-mail Template or for the GM+View. The actual pointer in the body of my GoldMine template looks like this:

<<file:Y:\GoldMine\HTML Templates\DJ Blue Contact Info.htm>>

When I use this template as my E-mail template for any client, it pulls all of the contacts information from the currently active record in GoldMine, and sends that information to the contact for review. They are asked to change any information that is not correct, and **Submit** the form to have the corrections made automatically back in my GoldMine. Alternatively, they may select that the information is correct as it stands, and I will receive an E-mail stating such, or they may even select to be removed from my data set. In this case I would also receive an E-mail requesting that they be removed from my GoldMine. Other than the E-mail to me, this step is not automatic for, I hope, obvious reasons.

The form, as I sent it to myself, is displayed in Figure 9-1 on the next page. Now, this form, could just as easily have been put on my web site, without the user information being pushed from GoldMine, and any end user could then have registered, updated their information, or even supplied their information, and asked to be removed from our data set with only slight modifications. Again, if the registrant were new to our data set, a record would be created automatically. If the registrant were in our data set, the information could automatically update their information. A removal request, would send us an

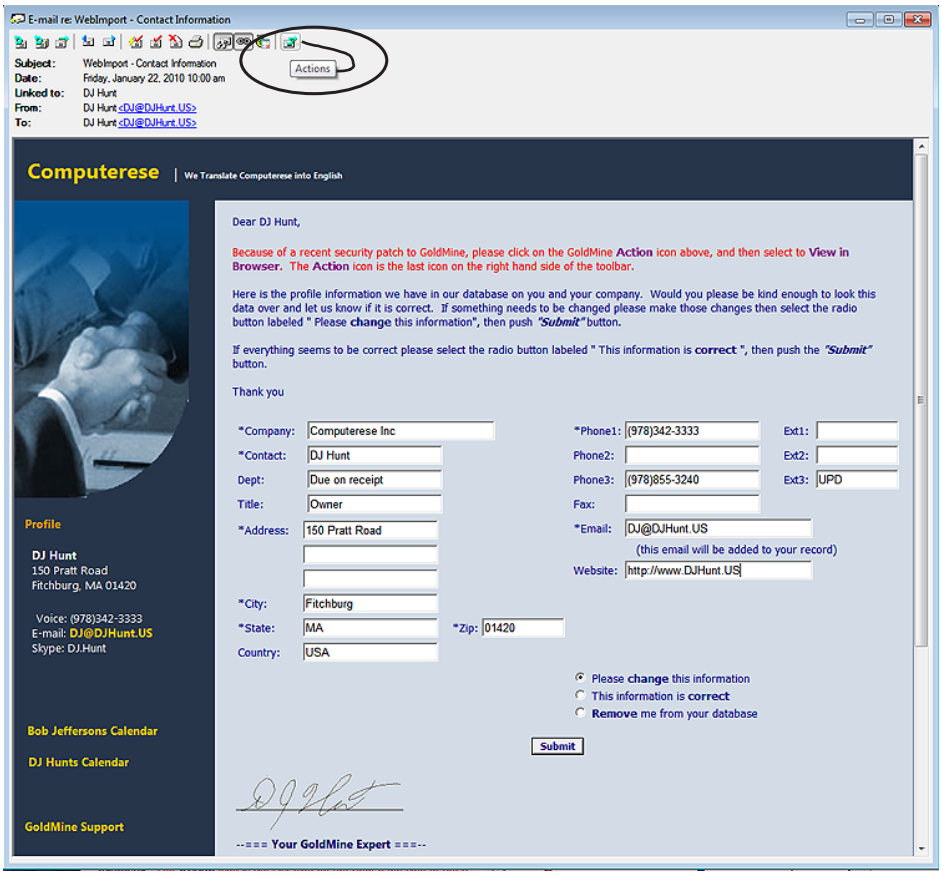


Figure 9-1

E-mail requesting to be removed. This action, is not automatic, and must be performed by the recipient of the message. I chose to review it as an E-mail template, as it is something that you could use in your practice as well, and as I said, it is easily converted to a web site form by just changing the duplicate checking statement for AccountNo to Email.

GM.ini

First, let's talk about the updatability of the GoldMine contact record. Having the ability to update a contact record, via a WebImport identified E-mail, was available as far back as in GoldMine 6. To use this ability, the GoldMine Administrator, must first tell GoldMine that they are willing to allow records to be updated, and specifically which **Contact1**/**Contact2** fields that they will permit to be updated via the WebImport message. This is accomplished through the global GoldMine instruction set, the **GM.ini**. I will show you an easier way to work with the GM.ini later on in this chapter, but for now, here are the necessary components along with an explanation.

Note

*This information is only indented/formatted for presentation purposes in this book. The actual **GM.ini** is modified via **NotePad** and must contain no formatting what-so-ever.*

In the GM.ini, the GoldMine Administrator must add a new section, as shown in my example here:

```
[WebImportOverwrite]
IAgree = I Want To Allow Fields To Be Updated By WebImport
Fields = Company, Contact, Department, Title, Address1, Address2, Address3, City, State, Zip,
         Country, Phone1, Phone2, Phone3, Fax, Ext1, Ext2, Ext3
```

The **[WebImportOverwrite]** section was newly added in GoldMine 6, and the first statement in this section leaves me a little baffled. Why would I even add this section if it were not my intent to update GoldMine records? However, the line is required **exactly** as is. That is spaces where I show you there are spaces, and no spaces in the variable, **IAgree**.

The **Fields =** statement, identifies those **Contact1**/**Contact2** fields that, you as the Administrator, will allow to be updated externally. Only those fields that are contained in this list will be considered to be updatable by the GoldMine application. Here we encounter a potential problem. The field list must be contained in one continuous line in the **GM.ini**. There is a 256 character limit to a single line in an ini file. Remember that we ran across that limitation when I was discussing the **Lookup.ini** earlier. Therefore, if you have many field names, and you wanted to be able to update a lot of fields through the WebImport, you would need to make certain that each line is under 256 characters. You may add consecutive **Field(x)** statements if you have more fields than can be included in a single statement.

Note

Had this been done using the GoldMine WebImport Tool, the Contact1/Contact2 field names would have all been in caps.

WARNING

*Some features of WebImport in GoldMine Premium now require that you utilize a **Password**. In fact, when using the built in tool in GoldMine Premium to modify the **GM.ini**, the **OK** button is disabled until you have entered a **Password**.*

WARNING

*The **Passwordx=** statement of the **[WebImpPassword]** section has limitations. The password may not contain spaces, and may not exceed 20 characters in length.*

*Although you may add additional passwords, as many as **999**, in the **GM.ini**, each **Instruction Set** may only contain **1** password. (See Process.asp later in this chapter)*

Here is an example of just such a situation:

```
[WebImportOverwrite]

IAgree=I Want To Allow Fields To Be Updated By WebImport

Fields=phone1, phone2, phone3, fax, ufg1, ufg2, ufg3, ufg4, ufg5, ufg6, ufg7, ufg8, ufg9, ufg10, ufingoals, umg1, umg2, umg3, umg4, umg5, umg6, umg7, umg8, umg9, umg10, umgtgoals, uredux, ulnlife, upcomm, upcont, source, umaxdown

Fields1=urlt,ucrdrpt, uinchg,uinchgexp, uastchg, uastchgexp, udbchg, ubtrm6, udbcrd1, udbcrd2, udbcrd3, udbcrd4, udbcrd5, udbcrd6, udbpurp1, udbpurp2, udbpurp3, udbpurp4, udbpurp5, udbpurp6

Fields2=upaid, upaidexp, uregls3, uregls5, uregls10, ucpa, ucpar, ucfp, ucfpr, uins, uinsr, urltr, ubnk, ubnkr, urefneeds, uskip, uothermtg, uotherdebt, urinsco, email, notes
```

If you are planning on passing a password via your asp, then you must make sure that GoldMine is prepared to receive that password. The **GM.ini** section, and statements for that might be something like:

```
[WebImpPassword]

Password1 = ShowBoat
Password2 = Musical
...
Password999 = Show
```

Now I will take a look at the form that I will be using, the **Contact_Info.htm** form. As I eluded to earlier, this could just as easily have been a form on a web site as it is a form in our E-mail message.

Contact_Info.htm

The **Form Action** is an extremely important part of the form HTML code as it tells the action to be taken when the user clicks on the **Submit** button of the form.

```
<form action="http://DJ-Hunt.com/DWSite/Asps/Process.asp" method=POST name=form1 OnSubmit="return validateForm(this)">
```

This code could actually be truncated to:

```
<form action="http://DJ-Hunt.com/DWSite/Asps/Process.asp" method=POST>
```

Note

*Throughout this chapter I will be referring to documents that are included with the downloadable edition of this book. You may want to open those up in **NotePad** to follow along.*

Here is where I instruct you how the form action is to occur, and the method of the form action. In this case I am pointing the action to an asp file located on my site which will process the received form information into an E-mail message, and then send it to a GoldMine user. I will discuss the asp file in the next section of this chapter. The method that I have chosen, is the **POST** method. If the method of **POST** is not entered, then the default method of **GET** is employed. The **GET** method is not as secure as all of the data is exposed when it is added to the end of the URL. Also, our response asp is set up to read from a **POST**. It is very important that the **Form**, and **Response** relationship is maintained.

Another important element of the Contact_Info.htm is that each field has a name and, in this case, each field has a default value pulled from the GoldMine contact record. On a web form, these fields would not have a default value, and the user would be expected to supply all of the information. Here is some typical HTML code for a field on my form:

```
<input name="address1" type="text" value="&lt;&lt;&amp;address1&gt;&gt;" >
```

On **Submit**, the **Address1** name will be passed with the present field value to the response asp on my site for processing.

That is basically it for the **Form** portion of the **Form/Response** set. Yes there is a lot of formatting, and script on the form that I have supplied you. Make a copy of that form, and play around with it in your favorite HTML editor. However, there is one thing that I want to make you cognizant of, in this form I am passing two GoldMine fields as hidden fields, and it is these fields that will be utilized for matching purposes. If using this form on a Web Site, this information would not be readily available, and you would want to use something like the E-mail Address for matching purposes. They are coded thusly in my form:

```
<input name="accountno" type="hidden" value="&lt;&lt;&amp;accountno&gt;&gt;">
<input name="email" type="hidden" value="&lt;&lt;&amp;emailaddress&gt;&gt;">
```

Process.asp

Before I begin to analyze this asp, I want to dump a few prerequisites on you. Your ISP must provide some sort of mail service component, and most do already. In my case, I use the CDO mail component, and that works out quite well from my site host (Network Solutions).

I instantiate the object in my asp via:

```
Set WebImport = Server.CreateObject("CDO.Message")
```

Not only must I bundle the fields that the user responds with, but I must also bundle our processing instructions. I would like to list the various instructions that could be passed back to GoldMine for processing. Each line of the instructions, and data for that matter, must be followed by a Carriage Return and Line Feed. **vbCRLF** is Visual Basic predesignated value for a Carriage Return and Line Feed.

Note

*Please note that the instructions are passed surrounded by quotes ("). If you were building this string with concatenated elements, you would need to add to both sides of the string **chr(34)** which is the quote (").*

The instruction set **must** begin with a section designator:

```
WebImport.TextBody = "[Instructions]" & vbCRLF
```

In the above example, I am concatenating a string, and placing its value into the **WebImport.TextBody** property. That was for you programmers. For you non programmers, I'm stuffing a variable with a string of characters.

Some of the instructions that one could pass after the section designator are:

Sending a **Password** if GoldMine is setup to look for one, and it should be in GoldMine Premium v9:

```
WebImport.TextBody = WebImport.TextBody & "Password=YourPassword" & vbCRLF
```

Telling GoldMine **Not to Import** any data as you would like an external application to do it instead:

```
WebImport.TextBody = WebImport.TextBody & "ImportData=0" & vbCRLF
```

Telling GoldMine to **Check for the Duplicates** based upon, and these can be compounded:

```
WebImport.TextBody = WebImport.TextBody & "DupCheck=Email" & vbCRLF
WebImport.TextBody = WebImport.TextBody & "DupCheck1=Contact" & vbCRLF
WebImport.TextBody = WebImport.TextBody & "DupCheck2=Accountno" & vbCRLF
```

WARNING

You may either pass:

DupLogic=AND

or

DupLogic =OR

You should never pass both.

If you are passing more than a single **DupCheck** instruction, you must tell GoldMine that you want those DupChecks handle as if they were in an **AND** or an **OR** statement:

```
WebImport.TextBody = WebImport.TextBody & "DupLogic=AND" & vbCRLF
```

or

```
WebImport.TextBody = WebImport.TextBody & "DupLogic=OR" & vbCRLF
```

Telling GoldMine to **Launch an External Application** from within the WebImport:

```
WebImport.TextBody = WebImport.TextBody & "Run= C:\WinNT\NotePad.exe" & vbCRLF
```

WARNING

Statements must be one continuous line unless programmatically broken. Any wrapped lines shown here are only for eBook readability.

Telling GoldMine to **Create an E-mail** for a New/Dupe record, and this too may be compounded as well:

Syntax:

```
"OnNewSendEmail=<<UserID>>, <<ActvCode>>, <<Reference>>"

WebImport.TextBody = WebImport.TextBody & "OnNewSendEmail=DJ, NEW, New WebImport" & vb-
        CRLF
WebImport.TextBody = WebImport.TextBody & "OnNewSendEmail1=DJ, NEW, New WebImport" & vb-
        CRLF
WebImport.TextBody = WebImport.TextBody & "OnDupSendEmail=DJ, OLD, Old Web Import" & vbCRLF
WebImport.TextBody = WebImport.TextBody & "OnAnySendEmail=DJ, ANY, Any Web Import" & vbCRLF
```

Telling GoldMine to **Save a Copy** of the imported message in the **ContHist** table linked to this **Contact** record:

```
WebImport.TextBody = WebImport.TextBody & "SaveThis=Web import file" & vbCRLF
```

WARNING

Spaces in the Track name used to cause issues in prior versions of GoldMines WebImport. If you are seeing that the proper Track names are not being assigned via the WebImport, try renaming your Tracks, removing all spaces.

Telling GoldMine to **Attach a Track** to a New/Dupe record, and this statement may be compounded:

Syntax:

```
"OnNewAttachTrack=<<Track Name>>, <<Attaching UserID (optional)>>"
```

WARNING

GoldMine Premium requires that you pass a WebImport ***Password*** *in order for it to attach any* ***Automated Process*** *to a* ***Contact*** *record.*

```
WebImport.TextBody = WebImport.TextBody & “OnNewAttachTrack=WEB Lead” & vbCRLF
WebImport.TextBody = WebImport.TextBody & “OnDupAttachTrack=WEB Lead” & vbCRLF
WebImport.TextBody = WebImport.TextBody & “OnDupAttachTrack1=WEB LeadA” & vbCRLF
WebImport.TextBody = WebImport.TextBody & “OnDupAttachTrack2=WEB LeadB” & vbCRLF
```

That pretty much wraps up the available instructions as I know them. On the next few pages I will provide you with my entire sample asp file. I will comment on items that you will need to change.

```
<!DOCTYPE HTML PUBLIC “-//W3C//DTD HTML 4.0 Transitional//EN”>

<html>
   <head>
      <title>Process Page</title>
   </head>

   <body>

Begin script
   <%
```

I am just putting the form submitted data into variables in this section of the script. This step is not necessary, however, it does add to the readability of the code.

```
f_accountno = Request.form(“AccountNo”)
f_company = Request.form(“Company”)
f_contact = Request.form(“Contact”)
f_department = Request.form(“Department”)
f_title = Request.form(“Title”)
f_address1 = Request.form(“Address1”)
f_address2 = Request.form(“Address2”)
f_address3 = Request.form(“Address3”)
f_city = Request.form(“City”)
f_state = Request.form(“State”)
f_zip = Request.form(“Zip”)
f_country = Request.form(“Country”)
f_phone1 = Request.form(“Phone1”)
f_ext1 = Request.form(“Ext1”)
f_phone2 = Request.form(“Phone2”)
f_ext2 = Request.form(“Ext2”)
f_phone3 = Request.form(“Phone3”)
f_ext3 = Request.form(“Ext3”)
f_fax = Request.form(“Fax”)
f_email = Request.form(“Email”)
f_addedemail = Request.form(“AddedEmail”)
f_website = Request.form(“Website”)
f_change = Request.form(“R1”)
f_correct = Request.form(“R1”)
f_remove = Request.form(“R1”)
```

If user has selected **CHANGE**:

```
If f_change = “Change” then

   Set WebImport = Server.CreateObject(“CDO.Message”)

   WebImport.Subject = “Changes to contact record: “ & f_contact

   WebImport.From = f_contact & “<” & f_email & “>” ‘ Specify sender’s address

   WebImport.To = “{$GM-WEBIMPORT$}<DJ@DJHunt.US>”
```

WARNING

Statements must be one continuous line unless programmatically broken. Any wrapped lines shown here are only for eBook readability.

Create the **INSTRUCTIONS** section of your WebImport:

```
WebImport.TextBody = “[Instructions]” & vbCRLF
WebImport.TextBody = WebImport.TextBody & “DupCheck=Accountno” & vbCRLF
WebImport.TextBody = WebImport.TextBody & “OnNewSendEmail=DJ, NEW, New Web Import” & vbCRLF
WebImport.TextBody = WebImport.TextBody & “OnDupSendEmail=DJ, OLD, Old Web Import” & vbCRLF
WebImport.TextBody = WebImport.TextBody & “SaveThis=Web import file” & vbCRLF
```

Creating your **DATA** section of WebImport:

```
WebImport.TextBody = WebImport.TextBody & “[Data]” & vbCRLF
WebImport.TextBody = WebImport.TextBody & “AccountNo=” & f_accountno & vbCRLF
WebImport.TextBody = WebImport.TextBody & “Company=” & f_company & vbCRLF
WebImport.TextBody = WebImport.TextBody & “Contact=” & f_contact & vbCRLF
WebImport.TextBody = WebImport.TextBody & “Department=” & f_department & vbCRLF
WebImport.TextBody = WebImport.TextBody & “Title=” & f_title & vbCRLF
WebImport.TextBody = WebImport.TextBody & “Address1=” & f_address1 & vbCRLF
WebImport.TextBody = WebImport.TextBody & “Address2=” & f_address2 & vbCRLF
WebImport.TextBody = WebImport.TextBody & “Address3=” & f_address3 & vbCRLF
WebImport.TextBody = WebImport.TextBody & “City=” & f_city & vbCRLF
WebImport.TextBody = WebImport.TextBody & “State=” & f_state & vbCRLF
```

```
WebImport.TextBody = WebImport.TextBody & "Zip=" & f_zip & vbCRLF
WebImport.TextBody = WebImport.TextBody & "Country=" & f_country & vbCRLF
WebImport.TextBody = WebImport.TextBody & "Phone1=" & f_phone1 & vbCRLF
WebImport.TextBody = WebImport.TextBody & "Ext1=" & f_ext1 & vbCRLF
WebImport.TextBody = WebImport.TextBody & "Phone2=" & f_phone2 & vbCRLF
WebImport.TextBody = WebImport.TextBody & "Ext2=" & f_ext2 & vbCRLF
WebImport.TextBody = WebImport.TextBody & "Phone3=" & f_phone3 & vbCRLF
WebImport.TextBody = WebImport.TextBody & "Ext3=" & f_ext3 & vbCRLF
WebImport.TextBody = WebImport.TextBody & "Fax=" & f_fax & vbCRLF
WebImport.TextBody = WebImport.TextBody & "Email=" & f_addedemail & vbCRLF
```

I need to break up the **Process.asp** here to be able to explain a bit about **Notes** even though the **Contact_Info.htm** that accompanies this ASP does not include any Notes, there will be cases where you need to compensate for them. Notes sent via WebImport must be broken up into chunks of 255 characters. I show you a sample of syntax below.

We'll have to assume that I have set up a variable **f_Notes** to capture the **Form** field **Notes**, and now the script must begin parsing same:

```
If len(trim(f_Notes)) > 0 then
   WebImport.TextBody = WebImport.TextBody & "Notes=" & mid(f_Notes, 1, 255) & vbCRLF
End If

If len(trim(f_Notes)) > 255 then
   WebImport.TextBody = WebImport.TextBody & "Notes1=" & mid(f_Notes, 256, 255) &  vbCRLF
End If

If len(trim(f_Notes)) > 511 then
   WebImport.TextBody = WebImport.TextBody & "Notes2=" & mid(f_Notes, 512, 255) & vbCRLF
End If

If len(trim(trim(f_Notes)) > 766 then
   WebImport.TextBody = WebImport.TextBody & "Notes3=" & mid(f_Notes, 767, 255) & vbCRLF
End If
```

Creating a new **ContSupp** record via the WebImport:

```
WebImport.TextBody = WebImport.TextBody & "[ContSupp]" & vbCRLF
WebImport.TextBody = WebImport.TextBody & "cs1_RecType=P" & vbCRLF
WebImport.TextBody = WebImport.TextBody & "cs1_Contact=Web Site" & vbCRLF
WebImport.TextBody = WebImport.TextBody & "cs1_ContSupRef=" & f_website & vbCRLF
WebImport.TextBody = WebImport.TextBody & "cs1_Zip=0000"

On Error Resume Next
WebImport.Send

If Err <> 0 Then
  Response.Write "Error encountered: " & Err.Description
End If

Response.Write "Thank you for your help. Your changes will be made to our database."
```

User checked **CORRECT** option:

```
ElseIf f_correct = "Correct" then
 Set WebImport = Server.CreateObject("CDO.Message")
 WebImport.Subject = f_contact & " said no corrections required"
 WebImport.From = f_contact & "<" & f_email & ">" ' Specify sender's address
 WebImport.To = "DJ@DJHunt.US"
 WebImport.TextBody = f_contact & " states that as of " & Date
                      & " their contact information is correct"
On Error Resume Next
WebImport.Send

If Err <> 0 Then
  Response.Write "Error encountered: " & Err.Description
End If

Response.Write "Thank you for your help"
```

User checked the **REMOVE** option:

```
Else
 Set WebImport = Server.CreateObject("CDO.Message")
 WebImport.Subject = f_contact & " asked to be REMOVED"
 WebImport.From = f_contact & "<" & f_email & ">" ' Specify sender's address
 WebImport.To = "DJ@DJHunt.US"
 WebImport.TextBody = f_contact & " states that as of " & Date
                      & " their contact information should be REMOVED"
```

WARNING

*The GoldMine WebImport is incapable of properly handling imbedded carriage returns and line feeds when using the **ini** type of WebImport, and will stop processing the **Notes** at the first instance of a **vbCRLF**. As you cannot prevent users from using the carriage return, you may want to remove the **vbCRLF**, via your script, from the **Notes** string before parsing it into the **[Data]** section.*

Note

*I cover the **Notes** here, not because I use them via this E-mail WebImport as I don't, but because you could easily convert the E-mail WebImport into a Website WebImport into GoldMine where you would want to capture Comments and/or Notes. The **Process.asp** could almost be utilized as is, while the **Contact_Info.htm** need only be modified to remove the GoldMine feeds, and to fit your website theme.*

Note

***ContSupp** records are **not** updatable via the WebImport, hence, one must create a **New ContSupp** record via WebImport.*

*As of my last test, you could only create one **New ContSupp** record for each **ContSupp.Contact** type per WebImport. I.e. Don't try to create two **ContSupp.Contact = [Web Site]** during the same WebImport process.*

Your mileage may vary, but let me know if it does please, by E-mailing me at DJ@DJHunt.US.

Note

*Notice in the **Correct** and **Remove** sections that the **WebImport.To =** is different than our normal WebImport. To statement as these two only need to be messages to the GoldMine UserID, and are not really a WebImport.*

WebImport:

```
WebImport.To = "{$GM-WEBIMPORT$}
               <DJ@DJHunt.US>"
```

E-mail Message:

```
WebImport.To = "DJ@DJHunt.US"
```

```
On Error Resume Next
WebImport.Send

If Err <> 0 Then
  Response.Write "Error encountered: " & Err.Description
End If

Response.Write "Thank you for your help. You will be removed from our database."

End If
%>
</body>
</html>
```

E-mail Rule

The last item that I needed to take care of is an **E-mail Rule** for this recipient, just in case. The E-mail Rule in GoldMine under this recipients account, in this case **DJ@DJHunt.US**, would be something on the order of:

Incoming Rule : **WebImport**
IF
Subject Contains *$GM-WEBIMPORT$* **OR**
To Contains *$GM-WEBIMPORT$*
THEN
WebImport

...as show in the GoldMine E-mail Rule Center English Script.

Note

If you purchased the printed copy of this book, then make sure that you send me the receipt as you are entitled to a free pdf copy of this book as well with all of the related sample files.

WebImport - Script Generator

This, then, just about wraps up our entire discussion on **WebImport**. WebImporting, and updating is an absolutely great feature within GoldMine.

In its continuing effort to add functionality to GoldMine, FrontRange had added a new utility to GoldMine as of the GoldMine 6.6 product build, and it is known as **Web Import Profiles**. This utility allows one to create, via a GUI, a WebImport entry form along with its associated script file. The forms can be used directly from the FrontRange web site on a temporary basis, and/or can be e-mailed to the creator for modification to be incorporated, let's say, into the users theme.

I would like you to look at this new utility as I do, and I will walk you through the creation of a single Web Import Profile. Click on

Web
Setup Web Import...

(don't you just love the way that FrontRange keeps varying its usage of the term **Web Import/WebImport** throughout GoldMine?) to bring up the dialog form shown here in Figure 9-2.

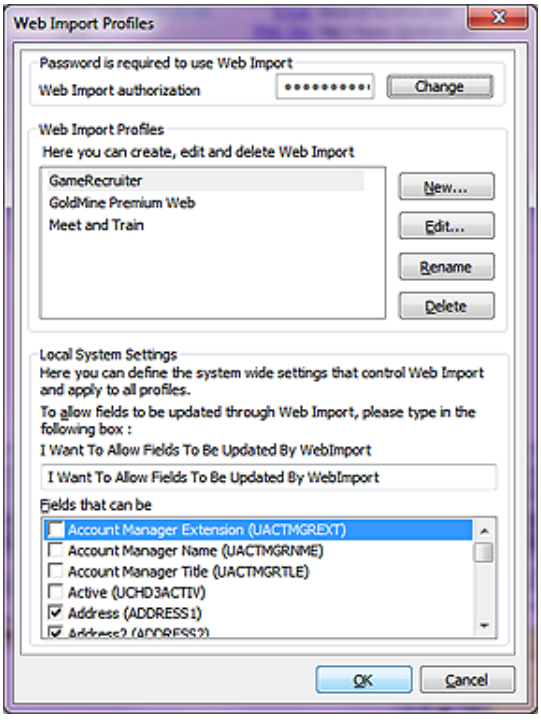


Figure 9-2

Figure 9-2 shows a changed dialog form when compared to the dialog form in previous editions of GoldMine as the **More >>** button no longer exists. One form size fits all now a days.

I'm going to first cover the **Password is required to use Web Import** frame where there is a single field, and it is a required field if you wish to utilize the features of this dialog form. You must add a **Web Import authorization** password in order to enable this dialog form, and its many features. If this had been a first time use of this dialog form, then the word **Change** would have actually been the word **Save**. This dialog form read my GM.ini, hence, the dialog form is currently **Enable**. If the caption on the button had been **Save** then once the 1st letter of the password had been entered into the field, in this case **G** for **GoldMine**, all features would have been enabled automatically.

Note

*Please note, in GoldMine Premium, that you will not enable the **OK** button unless you have included a **...password:**. Although, passwords are only required to enable certain features, to utilize the **Web Import Profiles** tool you must utilize a password.*

So why have a **Save** button at all for the password? That is an excellent question. You see the

developers are allowing you a way to configure your GM.ini via the GUI which is a very nice feature. After you have saved your password the **Save** button alters to a **Change** button as I had previously mentioned, and the ability to input another password is disabled. Various warnings are thrown at you when clicking upon these buttons.

Clicking upon the **Save** button will cause the below section to be incorporated into your GM.ini, and, although you have to add them by hand, I have also shown you how to add up to 999 addition passwords:

```
[WebImpPassword]
password=GoldMine
Password1 = ShowBoat
Password2 = Musical
...
Password999 = Show
```

Mind you, I have not actually tested whether the WebImport feature will honor these additional passwords as of this writing, however, this has always worked in the past, and I see no reason for the developers to have changed this feature.

Now I would like to discuss the **Local System Settings** frame. This component will manipulate your **GM.ini** as did the **Password is required to use Web Import** frame component. Earlier in this chapter, I showed you the required settings in the GM.ini for the proper functioning of the WebImport when wanting to update fields in your GoldMine. With the advent of this GUI, FrontRange has made that job a whole lot easier.

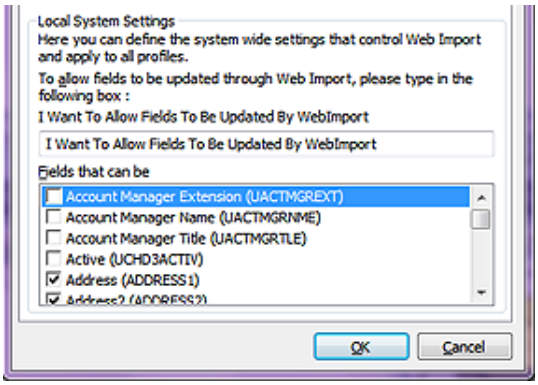


Figure 9-3

Unlike previous editions of this tool, you do not have to complete the field: **I Want To Allow Fields To Be Updated By WebImport** to enable the Fields that can be updated list. However, you must do this if you want any of your selected fields to be updatable. Go ahead, and type **I Want To Allow Fields To Be Updated By WebImport** into this field exactly as it is presented on the dialog form, and I do mean exactly. Caps where there are caps, and lower case where it is showing as lower case.

Having done that correctly, the fields from your **Contact1**/**Contact2** tables, those that are potential candidates for updating, can now be selected. You may simply select (☑) those **Fields that can be** updated by your incoming WebImport within your GoldMine.

These actions will then be represented in your GM.ini in the following manner:

```
[WebImportOverwrite]

IAgree = I Want To Allow Fields To Be Updated By WebImport

fields = ADDRESS1,ADDRESS2,ADDRESS3,PHONE2,USERDEF02,PHONE3,CITY,COMPANY,
CONTACT,KEY3,COUNTRY,UDOB,DEAR,EMAIL,EXT1,EXT2,EXT3,EXT4,FAX,KEY1,KEY2,LASTNAME,
SECR,KEY4,NOTES,PHONE1,UQBCUSTOMR,UQBVENDOR,USERDEF10,UCEXPIRE,UCHOURS,
SOURCE,UCSTART,STATE,KEY5,DEPARTMENT

fields1 = TITLE,USERDEF04,USERDEF07,USERDEF09,USERDEF01,USERDEF03,USERDEF05,
USERDEF06,USERDEF08,UWEBIMP,WEBSITE,ZIP,ULOGIN,UPW
```

Additionally, if you select more fields than can be handled by a single fields statement, GoldMine breaks the lines at the appropriate lengths, and inserts a continuation statement. This saves you a tremendous lot of character counting when typing your statements by hand. In fact, I use this tool exclusively as I don't have that many fingers, and toes to count that high. If doing this by hand, however, your statement might look like this:

```
fields = ADDRESS1,ADDRESS2,ADDRESS3,PHONE2,USERDEF02,PHONE3,CITY,COMPANY,
CONTACT,KEY3,COUNTRY,UDOB,DEAR,EMAIL,EXT1,EXT2,EXT3,EXT4,FAX,KEY1,KEY2,LASTNAME,
SECR,KEY4,NOTES,PHONE1,UQBCUSTOMR,UQBVENDOR,USERDEF10,UCEXPIRE,UCHOURS,
SOURCE,UCSTART,STATE,KEY5,DEPARTMENT
fields1 = ...
fields2 = ...
fields3 = ...
...
fields999 = ....
```

Note

*You may have noticed, a space per field is a premium in an ini statement, and that this tool has not included any spaces into the statement string within the **GM.ini**.*

Mind you I haven't tested the limitations of this statement area, however, going by similar rules for other sections this should work. If it is of any comfort to you, I have tested up to fields6 = for a client without issue.

One could simple click on the **OK** button, and the proper statements would be written to the GM.ini. No Web Form or ASP script would be created, but the GM.ini would be correctly formatted.

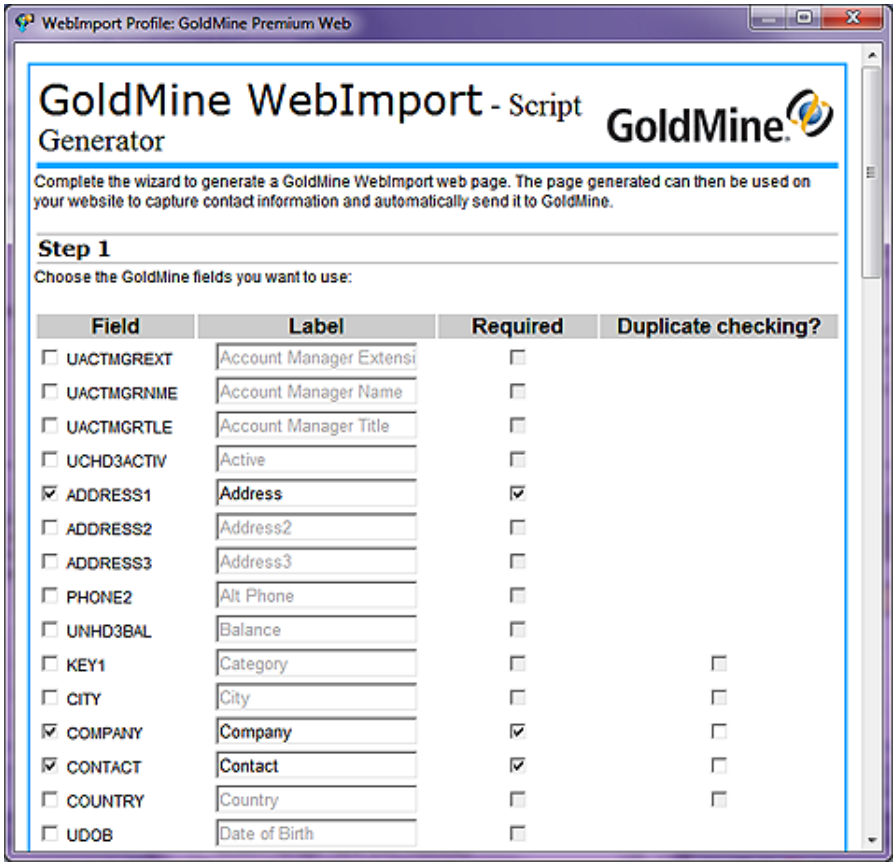


Figure 9-4

This is not my intent here, however, I am here to have the configurator develop a Web Form, and an ASP script file for this exercise. Returning now to the **Web Import Profiles** frame of Figure 9-2, and I quote, *"Here you can create, edit and delete Web Import profiles"*. Profiles, hmmm! More than one. Yes you could develop as many different profiles as you wish. My intent here is to show you, simply, how to create a single profile. Shall we continue (a rhetorical question, I know)?

Clicking on the **New...** button creates an entry in the profiles list with **New Profile** selected. Simply begin typing in a new profile name. I have entered **GoldMine Premium WebImport**. When you have completed entry in the profile name, hit the **Enter** key, and watch what develops, see Figure 9-4 above.

You will notice that some of the fields have been preselected for you already, and, of those, some have been preselected as being **Required**. As well, in the fourth column, some have been preselected for **Duplicate checking**?. Naturally, you can alter this form at will. You must select all of the fields that you wish to have contained on your Web Form. Of those fields, you must identify those that will need to be required fields. i.e. The user will not be able to **Submit** the form from your web site, unless there is information contained within those fields selected as required. Lastly, you should select, at least, the **E-mail Address** for duplicate checking. Remember, GoldMine will check the WebImport to see if a record already exists within your GoldMine based upon your choice(s) for duplicate checking?. Depending on whether you are allowing the updating of the record when a duplicate is found, GoldMine will either do nothing or update the fields with the newer WebImport information when the WebImport is processed by GoldMine.

One thing that I haven't explained, is the second column on this form, the **Label** column. This column is where you may change the Web Form field display label that is associated with said field. You may type into the field in this column to change the associated label.

Once completed, as cannot be seen at the bottom of Figure 9-4, and similar to those shown in Figure 9-5 on the next page, there are buttons to either **Reset the Form**, **Next >**, or **Cancel** the activity.

Note

You may have noticed, Figure 9-2, that I have two profiles for my clients already created, as well as the profile that I am using for this book.

Note

Did you notice that the form, Figure 9-4, did not pick up on any of the fields which you had previously selected as updatable? One would think that if one wanted those fields to be updatable, that one would also want those fields on their webform. Seems logical to me.

Note

*As the **E-mail Address** is unique within the GoldMine default environment, this field alone may be sufficient enough for duplicate checking. The **E-mail Address** or the **AccountNo** information, if I have it, is all that I use for duplicate checking.*

WARNING

Though it appears that one can create a label of unlimited length on the webform, only about 31 characters will actually display.

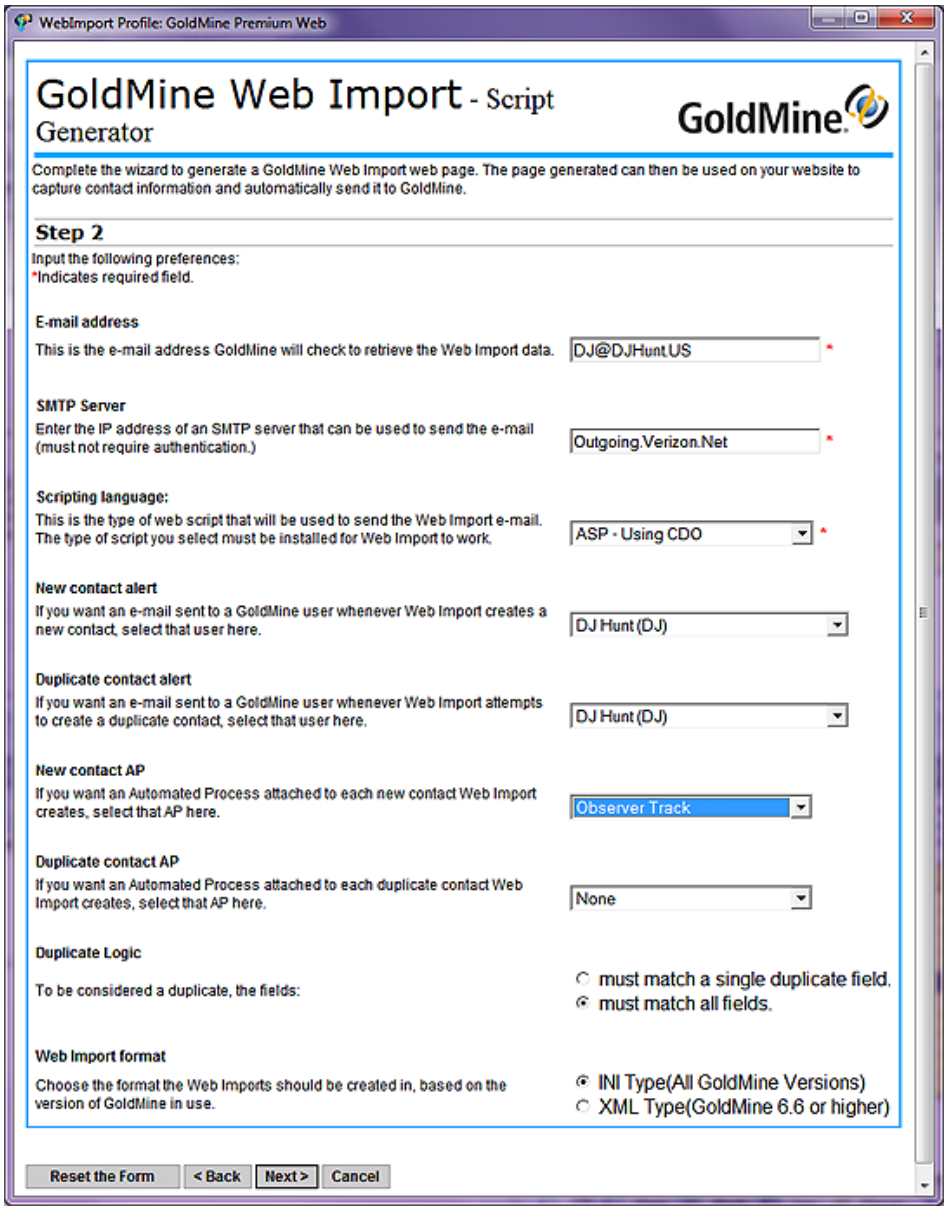


Figure 9-5

I have chosen to proceed with my development, so I have selected the **Next >** button. As a result, I am moved to **Step 2**, refer to Figure 9-5, where I am asked for additional options.

The astute reader will immediately notice that some of the fields are required as indicated by a red asterisk (*). These fields are the **E-mail address**, the **SMTP Server**, and the **Scripting Language:**. Let's take a look at these options one at a time, although most of them are self explanatory.

E-mail address - This is the E-mail address to which, the developed script, will be sending any WebImport Instruction Sets to, and is the E-mail address that you will want to set up in your GoldMine to Auto-Retrieve as well as set up a WebImport E-mail Rule against.

SMTP Server - This may be a little more difficult to come by for most users, as they state that the SMTP **must not require authentication**, while virtually all SMTP's today require authentication. This is the vehicle through which the script will attempt to send the Web Import Instruction Set. You might use **Outgoing.Verizon.Net**, however, one could as easily have used **203.196.185.43** if one knew the IP address of their SMTP. Actually, there are quite a few that still do not require authentication, but it is up to you to locate one. My particular Web Hosting service, Network Solutions, allows me to use **Server.CreateObject("CDO.Message")** so I don't even have any clue as to what the SMTP IP address is, and I don't need to know, as CDO takes care of all of that for me. Talk to your ISP.

Scripting language: - Can you say redundant? Why FrontRange puts an asterisk (* Required input) next to a field that contains a drop list, and can't possible be left blank, is beyond my imagination. However, having said that, this is where one selects that scripting language to be used to compile the web import instruction set. Personally, I use a variant of the default **ASP - Using CDO**, however, the other choices that you have available are:

ASP - Using ASPMail
ASP - Using CDONTS
ASP - Using CDO
PHP
PERL Using NET::SMTP

One or all may be possible on your server. PHP is fast becoming the scripting language de jour, while PERL is becoming passé. On the other hand, **ASP - Using CDNOTS** (Microsoft Collaboration Data Objects for Windows NT Server) is associated with the arrival of IIS for NT. Again, talk to your ISP to determine which would be best for your web server.

New contact alert - This is an item that I highly recommend that you complete. You will notice that the configurator reads your **UserID** table, and list the UserIDs here for your selection. If you don't have a UserID selected here, when GoldMine automatically reads the WebImport, and creates a new record, no one will know that a new record was added to the table unless they happen to trip over it. By notifying a user that a new record has been created, an alarmed activity is linked to the record that is created (see sidebar). No missing New Leads now.

Duplicate contact alert - Well what do you suppose this means? A duplicate, whether you choose to update or ignore, under automatic Web Import, goes unnoticed. As with the New contact alert, you may select a UserID and, upon duplicate, a Linked, Alarmed GoldMine Internal E-mail will be sent to the specified UserID (see sidebar).

New contact AP - Should you want an Automated Process attached to a New Contact record created by your WebImport, then you should select one here from your predefined processes. I always opt for my **Observer Process** as discussed in Chapter 10. If you have selected an AP to be attached to all New Contact records, as I'll discuss in Chapter 10, then the WebImport AP would attached it to the record in addition to the defined one. All of the available Automated Processes are read directly from your GoldMine, and presented here in the drop list. This is great if you are using an AP that automatically populates certain additional fields based on information contained in the WebImport or let's say you have a New Prospect Track.

Duplicate contact AP - Again we have the option of attaching an Automated Process, however, this time it is for when a duplicate WebImport has been received.

Duplicate logic - This is a radio button selection. Should you have selected multiple duplicate conditions, you must instruct GoldMine as to how to handle these duplicate conditions. You may choose one of the two possible options:

○ **must match a single duplicate field.** (translates to dbase **.OR.**)
◉ **must match all fields.** (translates to dBase **.AND.**)

The default selection is shown above, where my selected duplicate fields must all be matched to be considered a duplicate record, however, I usually choose the other option as I am matching on just the required field of E-mail Address. As you know, in the default configuration of GoldMine, the E-mail Address must be unique within GoldMine, hence, a single match here is sufficient for our needs. If you'll remember on my WebImport E-mail form, I used the AccountNo as a hidden field, and as my dupcheck for the WebImport. That is because I had the AccountNo available to send hidden in the E-mail message, however, someone entering information from your Web Form will not be privy to the unique GoldMine AccountNo value, hence, you must capture something else that is unique such as their E-mail Address or a CustomerID.

Web Import format - This is your final option on this page. Again we have a two radio button choice only.

◉ **INI Type (All GoldMine Versions)**
○ **XML Type (GoldMine 6.6 or higher)**

Why change what has been working well for years. I accept the default setting as shown above. Again, XML is another language de jour, and has been added to the GoldMine arsenal for just that reason.

I will now continue on with **Step 3** of the **Script Generator** by clicking upon the **Next >** button. You will notice that there is very little to be done on this next dialog form. All that one needs to do is select, or not, a template. Your choices are simple, with examples of each displayed to the right. Your

Note

Had you followed my suggestions in Chapter 3:

[GMAlarm]
*OnByDefault=ACTSO**M**DL*

The user may select any activity RecType to have that activity alarmed by default. GoldMine uses the following RecTypes:

A = Appointment
C = Call
T = Nex Action
D = To-do
***M** = Message (Internal)*
S - Sales Potential
O = Other
L = Literature Request

*The **M** choice would cause the **New contact alert** or the **Duplicate contact alert** to pop an alarm for the requested UserID.*

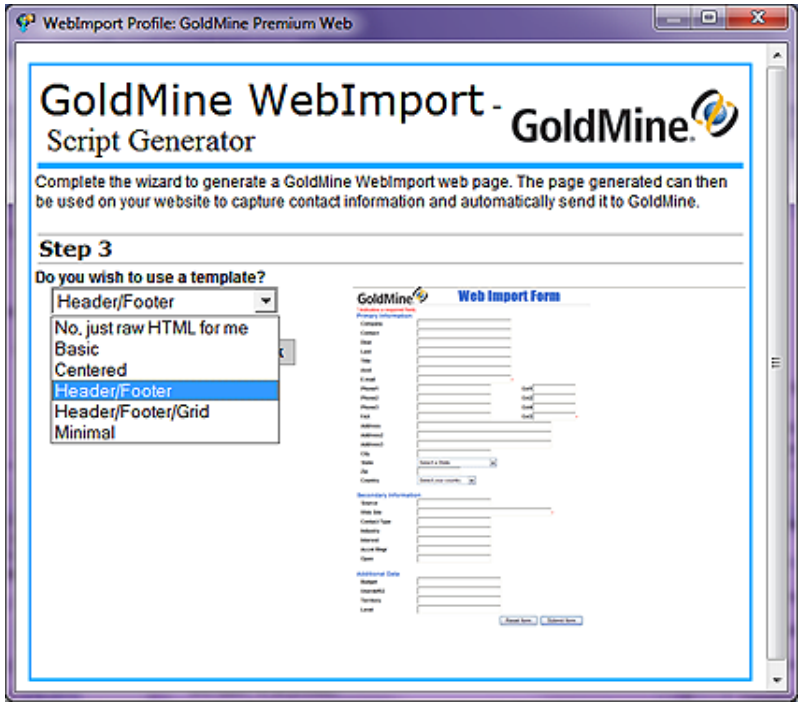


Figure 9-6

choices for Do you wish to use a template? are shown here in Figure 9-6. The default item in the drop list is **No, just raw HTML for me**.

All, except the default option, supply you with a form which you may later alter. The default option just provides the script, in your language of choice, for your reuse.

You will notice, in Figure 9-6, that I have selected, for presentation, the **Header/Footer** option. Simple logos, layout, and shading will be provided as displayed in the example. What is nice is that you can just use this form as the basis for your form in your Web Site theme schema, and you will not have to worry about passing the proper field names. This is, by itself, a blessing.

I can now, again, click on the **Next >** button to advance to **Step 4**, Figure 9-7.

Figure 9-7

You will notice that the only required information on the page is an **E-mail Address**. The E-mail Address to which you wish the script, and the form sent. The configurator will auto-populate the E-mail Address from the one which you entered earlier, however, you may change it if you wish the package to be sent to another E-mail Address.

You may click on the hotlink, **View WebImport Form**, to review the form as it has been designed, prior to having the package sent to you.

At this point, you may click to **Send the e-mail**, or **Save and Exit**. Additionally, you may choose to **Reset the Form**, to go **< Back** a page, or to simply **Cancel** the configurator entirely. I selected to **Send the e-mail**.

This result of my action can be seen in Figure 9-8. I received a nice little E-mail from **GoldMine WebImport <no-reply@frontrange.com>**, to which was attached the package that I had configured, **1064150406.zip**.

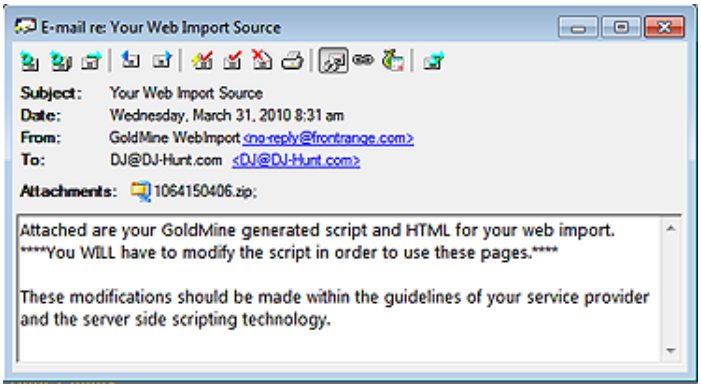


Figure 9-8

Additionally, the profile has been saved (although you may have to refresh your screen to see the **Profiles**), and may be edited at any time in the future, and/or I may create other profiles for other types of WebImport pages. In my opinion, this is a great time saver, hence, a great feature to GoldMine.

As the E-mail clearly states:

> **Attached are your GoldMine generated script and HTML for your web import.**
> ******You WILL have to modify the script in order to use these pages.******
>
> **These modifications should be made within the guidelines of your service provider and the server side scripting technology.**

Live within that, and you should have the basis for your WebImport.

DJ's Registration WebImport

Well, for a couple of books now I said that one could easily convert the **Contact_Info.html/Process.asp** into a Web Site document/script. I thought that, in this edition of the book, I would include a themeless version of my very own GoldMine Support Registration document and accompanying script. The templates have been included in your eBook download, and I will document it somewhat here. A full detailed description should not be necessary as we have already described this script previously.

For this exercise the templates are **GoldMine_Support_Registration.html** & **GoldMine_Support_Registration.asp** respectively. About the only thing that I can say on the GoldMine_Support_Registration.html document is that, as we don't have anything from within GoldMine on the Web Site for the user to select from, the **E-mail Address** field is a **Required** field. Since we use the default unique E-mail Address in GoldMine, we will be able to use the E-mail Address for duplicate checking via the WebImport. Also, a few of the fields are prepopulated with the most common value, for instance: **Country:** is prepopulated with **USA**, yet is editable. The **POST** method of this form points to the GoldMine_Support_Registration.asp on my web site, and that's pretty much all that I need to address on the HTML document.

Now let's look at the actual script, and you will find this is nothing more then a simplified form of the previously discussed Process.asp.

This begins the script section:

```
<%
```

This Creates the WebImport Object for use by this Web Page. Remember that I previous stated that I used the CDO SMTP capabilities:

```
Set WebImport = Server.CreateObject("CDO.Message")
```

Here is where you would put what you will want as the **Subject:** of the e-mail message that is being sent to your GoldMine:

```
WebImport.Subject = "<< GoldMine Support Registration >>"
```

This sets the **From:** value of the e-mail message that is being sent to GoldMine. Notice that I am stuffing the From value with the E-mail Address from the Web Page form that was submitted by the registrant (the **Required** field on the Form). This way the e-mail message appears to GoldMine as if it were sent directly by the registrant.

```
WebImport.From = Request.Form("EMAIL")
```

Here we set the **To:** value of the e-mail message. Please notice that it has the special coding that we will set in the GoldMine E-mail Rule that will designate this as a WebImport message. The actual E-mail Address should be that of the auto-retrieved e-mail account set up within your GoldMine for the WebImport process. Notice that I have remarked out the second or alternate recipient.

```
WebImport.To = "{$GM-WEBIMPORT$}<DJ@DJHunt.US>"
'WebImport.To = "{$GM-WEBIMPORT$}<Waldo@DJHunt.US>"
```

Note

*Placing an apostrophy (') as the first character of a statement is equivilent to **remarking** out that statement (turning the statement into a Comment line) in the script, hence, the color change.*

Here we begin the construction of the **Body:** value which includes both the **Instruction Set** and the **Data Set**.

Create the **Instruction Set** section of your WebImport:

```
WebImport.TextBody = "[Instructions]"  & vbCRLF
```

Define what GoldMine is to use for duplicate checking:

```
WebImport.TextBody = WebImport.TextBody & "DupCheck=Email"  & vbCRLF
```

Define the GoldMine Message to be created when GoldMine retrieves the WebImport with the syntax of <UserID, ActvCode, Reference>:

```
WebImport.TextBody = WebImport.TextBody & "OnNewSendEmail=DJ, NEW, New Web Import"&vbCRLF
WebImport.TextBody = WebImport.TextBody & "OnDupSendEmail=DJ, OLD, Old Web Import" & vbCRLF
```

Create the password for this specific WebImport:

```
WebImport.TextBody = WebImport.TextBody & "Password=GoldMine"  & vbCRLF
```

Attach an Automated Process, in this case only for those WebImports that are deemed New:

```
WebImport.TextBody = WebImport.TextBody & "OnNewAttachTrack=Observer, DJ" & vbCRLF
```

Create a History record in GoldMine that documents the contents of the WebImport message:

```
WebImport.TextBody = WebImport.TextBody & "SaveThis=WebImport File"  & vbCRLF
```

Here is the **Data Set**. These are the **Contact1**/**Contact2** fields which are to be updated or populated:

```
WebImport.TextBody = WebImport.TextBody & "[Data]" & vbCRLF
WebImport.TextBody = WebImport.TextBody & "Contact="&Request.Form("NAME") & vbCRLF
WebImport.TextBody = WebImport.TextBody & "Source="&Request.Form("Source") & vbCRLF
WebImport.TextBody = WebImport.TextBody & "Company="&Request.Form("COMPANY") & vbCRLF
'WebImport.TextBody = WebImport.TextBody & "Key2="&Request.Form("Industry") & vbCRLF
'WebImport.TextBody = WebImport.TextBody & "Key3="&Request.Form("Market") & vbCRLF
WebImport.TextBody = WebImport.TextBody & "Address1="&Request.Form("ADDRESS1") & vbCRLF
WebImport.TextBody = WebImport.TextBody & "Address2="&Request.Form("ADDRESS2") & vbCRLF
WebImport.TextBody = WebImport.TextBody & "City="&Request.Form("CITY") & vbCRLF
WebImport.TextBody = WebImport.TextBody & "State="&Request.Form("STATE") & vbCRLF
WebImport.TextBody = WebImport.TextBody & "Zip="&Request.Form("ZIP") & vbCRLF
WebImport.TextBody = WebImport.TextBody & "Country="&Request.Form("COUNTRY") & vbCRLF
WebImport.TextBody = WebImport.TextBody & "Phone1="&Request.Form("DAYPHONE") & vbCRLF
WebImport.TextBody = WebImport.TextBody & "Phone2="&Request.Form("NIGHTPHONE") & vbCRLF
WebImport.TextBody = WebImport.TextBody & "Ext1="&Request.Form("EXT1") & vbCRLF
WebImport.TextBody = WebImport.TextBody & "Fax="&Request.Form("FAXPHONE") & vbCRLF
WebImport.TextBody = WebImport.TextBody & "Email="&Request.Form("EMAIL") & vbCRLF
```

As this is an **ini** type of WebImport, we are using these functions to divide the user input Web Form **Comments** into statements existing of 255 characters in length each:

```
If len(trim(Request.Form("COMMENTS"))) > 0 then
   WebImport.TextBody = WebImport.TextBody & "Notes=" & mid(Request.Form("COMMENTS"),1,255)
   & vbCRLF
End If
```

```
If len(trim(Request.Form("COMMENTS"))) > 255 then
   WebImport.TextBody=WebImport.TextBody&"Notes1="&mid(Request.Form("COMMENTS"),256,255)
   & vbCRLF
End If
If len(trim(Request.Form("COMMENTS"))) > 511 then
   WebImport.TextBody=WebImport.TextBody&"Notes2="&mid(Request.Form("COMMENTS"),512,255)
   & vbCRLF
End If
If len(trim(Request.Form("COMMENTS"))) > 766 then
   WebImport.TextBody=WebImport.TextBody&"Notes3="&mid(Request.Form("COMMENTS"),767,255)
   & vbCRLF
End If
```

These fields we are stuffing with fixed data, and this information **does not** come from the Web Form:

```
WebImport.TextBody = WebImport.TextBody & "uWebImp=Y" & vbCRLF
WebImport.TextBody = WebImport.TextBody & "MergeCodes=ME" & vbCRLF
WebImport.TextBody = WebImport.TextBody & "CreateOn="& Date() & vbCRLF
WebImport.TextBody = WebImport.TextBody & "udRun=" & Date() & vbCRLF
WebImport.TextBody = WebImport.TextBody & "ucRunTime=06:00" & vbCRLF
```

Insert a blank line after the end of the **Data Set**:

```
WebImport.TextBody = WebImport.TextBody&" "& vbCRLF
```

Here is the start of the Contact Supplemental Data. You need to number each of the following of each section. For example, all of the cs1_ would go together and then cs2, etc...

```
if len(trim(Request.Form("URL"))) > 0 then
   WebImport.TextBody = WebImport.TextBody & "[ContSupp]" & vbCRLF
   WebImport.TextBody = WebImport.TextBody & "cs1_RecType=P" & vbCRLF
   WebImport.TextBody = WebImport.TextBody & "cs1_Contact=Web Site" & vbCRLF
   WebImport.TextBody = WebImport.TextBody & "cs1_ContSupRef="& Request.Form("URL") & vb-
   CRLF
end if

if len(trim(Request.Form("MSN"))) > 0 then
   WebImport.TextBody = WebImport.TextBody & "cs2_RecType=P" & vbCRLF
   WebImport.TextBody = WebImport.TextBody & "cs2_Contact=E-mail Address" & vbCRLF
   WebImport.TextBody = WebImport.TextBody & "cs2_ContSupRef="&Request.Form("MSN") & vb-
   CRLF
end if

'WebImport.TextBody = WebImport.TextBody & "cs3_RecType=P" & vbCRLF
'WebImport.TextBody = WebImport.TextBody & "cs3_Contact=Beyond Gold" & vbCRLF
'WebImport.TextBody = WebImport.TextBody & "cs3_ContSupRef=This is a test" & vbCRLF
'WebImport.TextBody = WebImport.TextBody & "cs3_Title=President" & vbCRLF
'WebImport.TextBody = WebImport.TextBody & "cs3_Dear=Dear" & vbCRLF
'WebImport.TextBody = WebImport.TextBody & "cs3_Notes=These are the notes for the special Detail"
   & vbCRLF

ON Error Resume NEXT
WebImport.Send
IF Err.number <> 0 THEN Response.Write( "There was an error while sending your message" & err.
   message )END IF
%>
```

Compare this with the previously mention Process.asp, and you'll notice that there is very little difference with even less difference between the two HTML forms where one is for an e-mail message, while the other is for a Web Site.

Go for it!

10

In This Chapter

Automated Processes is a tough chapter to write as a single chapter. I previously wrote an entire book, **Sales Force Automation with GoldMine 4.0**, as a blueprint guide to working with Automated Processes. Pretty much everything that I said in there then, still holds true for GoldMine Premium.

I still maintain that GoldMine is not going to tell you, or suggest to you, how to run your business nor are any of the GoldMine Partners, including myself. I further state, that one must have a process before one may begin to automate that process within GoldMine. A process should consist of more than a single task that you want to automate. A process should be a series of tasks that a user performs within GoldMine on a regular basis against any number of contact records.

In **Sales Force Automation with GoldMine 4.0**, still available as an eBook from **DJ@DJHunt.US**, I suggested that you create one Automated Process to watch over all the activity within GoldMine, and that Automated Process would assign the appropriate other Processes based upon a particular GoldMine activity. This particular process would be automatically attached to every **New** record created within GoldMine. The default Automated Process is identified in GoldMine through the corporate level GM.ini. A typical GM.ini statement identifying the Automated Process to be attached to all New records would be something like:

```
[GoldMine]
NewContactAP=9KUMSUA*XZ/XR#Y
```

Many readers of my past books have asked me to expand this chapter to include more examples. I plan to do that in this book again, however, I want to begin with a rehash of my former books.

Definitions

Automated Processes - a collection of **Processes** that could be scanned, and that are created/grouped by **UserID**.

Processes - a series of steps, **Events**, that are to be taken while the process is active. Each Process can be assigned certain attributes as shown here in Figure 10-1.

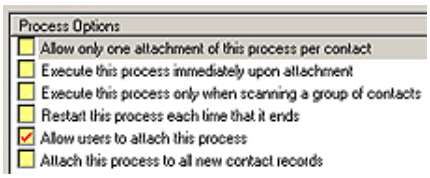


Figure 10-1

Of these options, which are all self explanatory, the last option is important. The last option may only be used once per GoldMine installation. As you read above, this option is written directly into the GM.ini file. A user will be warned, when selecting this option, if any other Process has already been defined for this GoldMine installation. I will suggest, later, that you select this option for the **Observer Process™** that we will be discussing in this chapter. This is the same exact paradigm that I employed when I wrote **Sales Force Automation with GoldMine 4.0**.

Event - the individual components of a **Process**, the ties of the processes, so to speak. Each **Event** possesses a **Description**, **Trigger**, and an appropriate **Action**.

Preemptive - Events that are analyzed each and every time that a **Process** is scanned, and that are always processed before any Sequential Events are even considered. This is an important concept to grasp. As you will see later in our Observer Process, I will make extensive use of Preemptive Events. Preemptive Events must have a Sequence Number between **10** and **99**.

Sequential - Events that are analyzed when a **Process** is scanned, however, each **Event** designated as a Sequential Event, may not be analyzed until the previous Sequential Event has been Triggered, and the associated Action has been executed. Sequential Events must have a Sequence Number between **100** and **999**.

Note

*As of GoldMine Premium 8.5, you may notice some terminology changes. For Instance: You can no longer **Remove** a **Track**, but, instead, you can now **Remove** a **Process**.*

***Tracks** have now been rebranded as **Processes**.*

Trigger - an activity that evaluates to a **True** causing the **Action** to be processed, or to **False** where the **Action** will not be processed.

Action - the activity that is performed once the **Trigger** has been evaluated, and when the evaluated **Trigger** returns a **True**.

The Observer Process™

I have long advocated the use of, and what I've trademarked as the **Observer Process™**. This Process will automatically be attached to each newly created contact record by GoldMine, and will be present during each scanning of the Automated Processes to test for activity against the Contact record to which this process has been attached. If one of the Events on the Observer Process Triggers, it should do nothing more than assign a new Process appropriate to the cause of the triggering.

The Observer Process will be made up of nothing more than Preemptive Events with one Sequential Event to hold the Process to the record so that it will never be removed from the record unless removed by an individual. Let's look at the creation of this Process.

I begin in the **Automated Process Center** as any user. In my case I selected the **(public)** UserID, however, I could just as easily used **DJ** or **MASTER**. I then clicked upon:

Tools
Automated Processes ►
Manage Processes...

Note

*Processes are automatically removed from the Processes tab by GoldMine when the last **Sequential Event** has been successfully executed.*

Note

***Process Codes** are used by GoldMine to create a display order for the **Processes** defined by a given user. This is an alpha ordering of the **Processes** in the **Automated Process Center**.*

Note

Those of you who will be Synchronizing your Automated Processes, and want your Remote-side to attach the Observer Process to all newly created records, will have to append the Remote-side GM.ini with the same statement that appears in the Server-side GM.ini. This statement, in my case, looks like:

[GoldMine]
NewContactAP=BOHRNWR$IY2UR#Y

Note

*You may want to consider the ☐ **Execute this process immediately upon attachment** option if you are attaching the Observer Process via the WebImport as discussed in a previous chapter.*

Figure 10-2

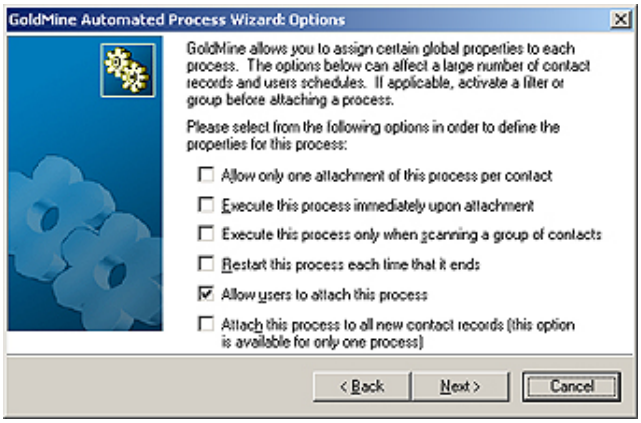


Figure 10-3

This will result in the dialog form as shown here in Figure 10-2. The **Process Name:** that I have established for this exercise is that of the **Observer Process**, while the **Process Code:** I set to **OBS**. The **Owner:** field is, in this instance, assigned to the **(public)** UserID. Clicking the **Next >** button brings up the dialog form shown in Figure 10-3.

As this is the Observer Process that will be attached to every Contact record in GoldMine, I do not want to allow more than one instance of this particular Process per Contact record. I will select to ☑ **Allow only one attachment of this process per contact**. Whereas, I do not want to ☐ **Execute this process immediately upon attachment**, so I will not select that option.

As well, I will not select to ☐ **Execute this process only when scanning a group of contacts**. This particular terminology is a bit confusing to some users. If you do not want the users to have the ability to scan this process for a single contact, select this option so the events will only be checked upon full contact set scans. I want to allow users to scan this process for single contacts, and have, therefore, not selected this option.

In case my **Hold Process**, discussed later, fails for any reason, I would want GoldMine to reattach this process to the Contact record. I have, therefore, selected to ☑ **Restart this process each time that it ends**. Further, I have opted to ☑ **Allow users to attach this process**, as there may be a need to do so, because of some previous failure(s). I want my users, in this case, to have the ability to assist me in my GoldMine maintenance tasks. Lastly, I have selected the option that may only be selected once per GoldMine installation, which is to ☑ **Attach this process to all new contact records (this option is available for only one process)**. Clicking on the **Next >** button brings us to the dialog form shown in Figure 10-04 on the next page.

Note

*I add a **Hold** Event to the Observer Process so that the Observer Process will never detach itself automatically from the record to which it is attached. Naturally, you may always detach the Observer Process by hand.*

Note

*As you are using the Builder, **AccountNo is Empty** will be all that you need to enter. I only supply the **Syntax:** here for your edification.*

I would ask you to click the **New** button here, and create the **Hold Process to Record** Event. I do this so that it will be available from the tree for creating new Events.

Event: Hold Process to Record
Type: Sequential
Sequence: 999

Trigger: dBASE Condition
Options:
English: AccountNo is Empty
Syntax: trim(contact1->accountno) < ‘ ‘ .or.”SQL_WHERE”=”(UPPER(c1.ACCOUNTNO) = ‘’ OR UPPER(c1.ACCOUNTNO) <= ‘ ‘ OR UPPER(c1.ACCOUNTNO) IS NULL)”

Attempt to trigger only once: ☐
Trigger Filter: ☐
Filter: None

Perform Action: Schedule Activity
Options: I usually schedule an Alarmed Other Action notifying DJ that the Observer Process was inadvertently removed.

When through, you can click on the **OK** button to be returned to the dialog form as shown here in Figure 10-4.

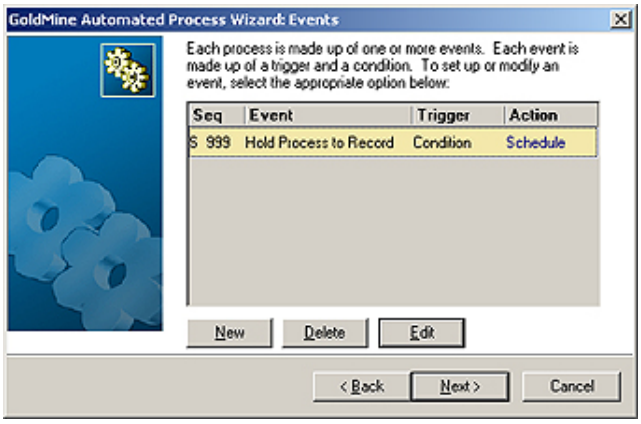


Figure 10-4

I will just click on the **Next >** button when I get back here as I will be creating the rest of the Events for the Observer Process later on in this chapter. That action brings us to the next dialog form from which I can do little more than click on the **Finish** button.

This will then create a tree similar to that shown, expanded, in Figure 10-5.

Note

It is important that you create your Processes prior to assigning them to the Observer Process for later dissemination.

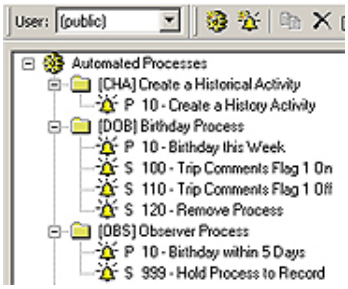


Figure 10-5

From here I can continue on with my Events. Let’s understand first what I will try to do here in my first Process. I am going to create a Birthday Process. This process will perform the appropriate actions for the Contact record.

For this to work as I would like, I will require the creation of a user defined field:

Field Name: uDOB
Description: Birthday
Field Type: Date, 8

Note

When naming Events, you want to be as descriptive as possible. You must know what the Event will accomplish when you look at attaching it to a record a year from now.

I would ask you to add this field to your GoldMine if you are to follow along on this example.

I can now continue on, and define our **Birthday Process**. Once I have tested the Birthday Process, I can then add a Preemptive Event to our Observer Process that will watch for certain conditions, and when Triggered, will attach our Birthday Process. If this sounds too simple, don’t be tricked into believing it. I have had the Automated Process logic laid out for years, and I’m writing this as a Step-by-Step match exactly as I have created these many times in the past. But let’s continue on.

Let’s now add the Birthday Process. You remember how to do this from my directions for adding the Observer Process, do you not? Create your own Birthday Process now with one bogus Preemptive Event (we will modify that later to meet the specific needs for this Process). Here is the Process criterion:

Process Name: Birthday Process
Process Code: DOB
Owner: (public)
Options:
☑ Allow only one attachment of this process per contact
☑ Execute this process immediately upon attachment
☐ Execute this process only when scanning a group of contacts
☐ Restart this process each time that it ends
☑ Allow users to attach this process
☐ Attach this process to all new contact records (this option is available for only one process)

Trick

You can create any dBase expression that you want to in the AP's if you follow these simple procedures:

*A. Create any expression using the **Expression Builder** shown later in Figure 10-8.*

*B. Click the **OK** button.*

*C. Click the **Options** button, shown later in Figure 10-7, to bring up the **Expression Builder**, Figure 10-8.*

*D. Highlight everything in the expression box, and begin typing your own expression by hand. **Do not** use the **Expression Builder** this time.*

*E. Click the **OK** button when finished.*

*If you have typed a valid expression, even if it is not against the **Contact1/Contact2** tables, which is all that the **Expression Builder** allows, your expression will be saved, and utilized.*

Tip

Notice that I have left sufficient space between my sequencing in case I later find out that I have to add more Events in between these Events. This will save a lot of reshuffling later on.

Note

You may want to add a Branch to Event such that if this Event doesn't fire, no E-mail Address, then to print a birthday document instead for postal mailing.

Note

*You may have noticed that I prefer to utilize the old dBase **[** an **]** braces where possible. I find that the **'** , when doubled, can be confused with the '' when using formatted text as I do in this book.*

Note

*Pay particular attention to my usage of a **Flag** byte to prevent the multiple firing of events on subsequent scans of the AP's. You may find this to be a most useful trick in your development of your AP's.*

When finished, select the bogus Preemptive Event from this tree. Now, right-click, and select **Properties...** from the local menu. Edit this Preemptive Event to be the same as the first Event that is shown below, and then right-click on that Event again, in the tree, only this time selecting **New...** from the local menu. Do this for the final two Events for this Process.

Here is the list of Events that should be created for your Birthday Process:

Event: Birthday this Week
Type: Sequential
Sequence: 100

Trigger: Detail Record
Options: Detail: E-mail Address

Trigger Filter: ☑
Filter: substr(ContSupp->Zip, 2, 1) = [1] .and. empty(left(Contact2->Comments, 1))

Action: E-mail message
Options: Select your predefined Birthday E-mail Template

Event: Trip Comments Flag 1 On
Type: Sequential
Sequence: 105

Trigger: Immediate
Options:

Trigger Filter: ☐
Filter:

Action: Update Field
Options: Field: Comments Expression: [1]+right(Contact2->Comments, 64)

Event: Trip Comments Flag 1 Off
Type: Sequential
Sequence: 110

Trigger: Elapsed Days
Options: 6

Trigger Filter: ☐
Filter:

Action: Update Field
Options: Field: Comments Expression: [ ]+right(Contact2->Comments, 64)

Event: Remove this Process
Type: Sequential
Sequence: 115

Trigger: Immediate
Options:

Trigger Filter: ☐
Filter:

Action: Remove Process
Options: Field: Expression:

And here is the verbose explanation of what we have just create in the Birthday Process.

I already know that the birthday criterion was met, or this Process would not have been added to the Contact record in the first place by the Observer Process. I'll discuss that later when I ask you to attach the Event to the Observer Process that will attach this, the Birthday Process. Further testing would be redundant. I first test for the existence of an E-mail Address, and then make sure that there is a Primary E-mail Address. I do the Primary E-mail Address check in the Trigger Filter of Event 100, a Sequential Event. As you may remember from the chapter titled **The Tables**, the second byte of the **Zip** field, when set to **1**, indicates that this E-mail Address is the Primary Contact E-mail Address. If this Contact record has a Primary E-mail Address, I want to send them my birthday E-mail Template. At the same time, I also make sure that our Flag position is empty so that I do not send repetative e-mails, and I do this via the addition of the **.and. empty(left(Contact2->Comments, 1))** clause.

My next Event in this Process, is our first Sequential Event, number 105, and is to **Update** the **Comments** field with a character (**1**). This will prevent this process from being re-added later today or tomorrow should the Automated Processes be scanned through another time. We will look at this **Flag** in the **Trigger Filter** of our Observer Process Event when the time comes. I use the Immediate trigger as there is nothing really to test for in this case as we just want the Flag to be updated immediately.

The next Event is Sequential Event number 110, and it is here that we hold the process in place for a period of time, in this case 6 days. This is the second way that we prevent the Observer Process from re-adding the Birthday Process. Remember, in our Process Options, that I asked you to select: ☑ **Allow only one attachment of this process per contact**? Well as long as the Birthday Process is attached to the Contact record, no other copy of the Birthday Process may be attached to the Contact record. Combine this with our Flag, and we have doubled our protection against duplication of activities. At the end of the 6 days, this Event will trigger, and the Action is to reset our flag field to be utilized next year by this process.

The last Event is to remove the Process. We don't like to rely on GoldMine removing the Process after the last Event has fired, although in theory and in practice, GoldMine removes the Process after the last Event has processed. As a safety precaution, however, I force the removal of the Process through Sequential Event, number 115 of the Birthday Process. It doesn't hurt to be overly cautious when programming your Automated Processes.

At this point you should fill in the Contact2->uDOB field with a valid birthday that falls within the next 5 days. You should be able to test the Birthday Process to assure that it is functioning as expected. Go to your Contact record in GoldMine, or any testable record, and click on the Processes tab. Assure that there is no other Birthday Process under this tab, and if there is, right-click and remove that Process. Right-click now under the Processes tab, and select **Attach a Process** from the local menu. The **Attach an Automated Process** dialog form will appear, and all of the processes contained within GoldMine will be listed, alphabetized by the Code (remember I suggested that you always employ codes, well there is your reason). Select the DOB Birthday Process for the list. Your process should begin its scan immediately. Remember, you selected the option: ☑ **Execute this process immediately upon attachment**. You should see the E-mail departing immediately unless you had selected to queue your message. You should notice that the 1st byte of the Comments field is displaying a **1**. You should see the Birthday Process under the Processes tab with the next Event registering as **Trip Comments Flag 1 Off**.

Tip

*In the **Conclusion** section of this chapter, I will discuss how to turn on a **Process Monitor** feature in GoldMine that will display the Step-by-Step triggering of Events.*

Let me say, right here and now, that I would never advocate the use of Automated Processes for a single Event Process. Processes should consist of a series of events that are to be processed in a sequential and logical order. True, I could have done all that I had just done in a single Process, however, then you are not looking at the whole picture, the future of your Automated Processes. Each Process has a finite number of Preemptive and Sequential Events, and this, alone, promotes our philosophy of using an Observer Process to watch for events, and then fire the appropriate Process based on the Event that is being triggered. It also allows you to adhere to our **Tip** (see sidebar). Solve each Process individually, and then once solved, execute it from the Observer Process. This whole strategy makes excellent sense in a corporate environment where one could have hundreds of processes.

Tip

Do not attempt to solve the whole puzzle in one sitting. Take little pieces, and solve those. Eventually the whole picture will come into focus.

All of this has prepared us to begin adding an Event to our Observer Process that will attach the Birthday Process at the appropriate time. I am going to ask the Observer Process to watch for a birthday falling within the next 5 days, and if it does trigger on this condition, then to have it add the Birthday Process and scan it immediately.

Right-click on the **Observer Process**, **S 100 - Hold Process to Record**, and select **New...** from the local menu. This will bring us into the GoldMine **Automated Process Event Wizard** as shown here in Figure 10-6.

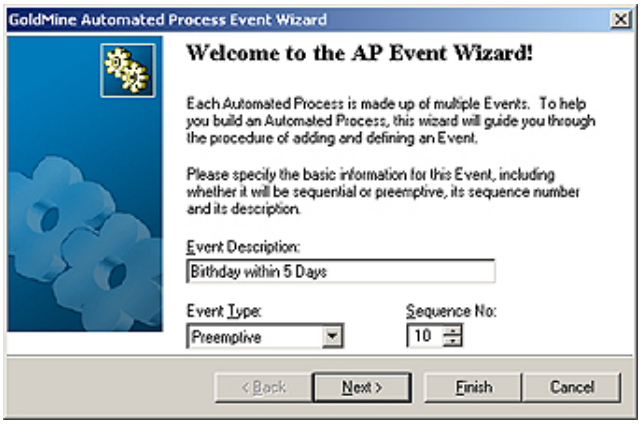


Figure 10-6

Notice that I have inserted **Birthday Within 5 Days** into the **Event Description:**, and that I have changed the **Event Type:** to **Preemptive** while setting the **Sequence No:** to 10. Click on the **Next >** button to move to the next page of the Wizard.

Note

*You may be asking yourself: **Where did DJ come up with the 5 days?***

I make the ascertion that the AP's are not scanned over the weekends, or weekends with holidays preceeding or following them. By utilizing a 5 day frame, I believe that I have caught most of the Birthdays that would have otherwise been missed when not scanning the AP's over the weekends.

I do hope that you find my logic to be solid.

Note

***Preemptive** events must have a **Sequence No:** between **10** and **99**. In theory, **1** thru **9** work as well, but I have witnessed Event ordering difficulties when using those numbers.*

Figure 10-7, presented on the next page of this book, shows the next page in this Wizard. Notice that I have selected **dBase Condition** to **Trigger on:**. Next I would ask you to click on the **Options** button to produce the dialog form shown in Figure 10-8, well almost, refer to the **Trick** that I talked about in a sidebar note on the previous/next pages.

This particular expression, as shown in Figure 10-8 on the next page, could have been entered using the Expression Builder, however, there will be cases where you will need/want to create ex-

Trick

You can create any dBase expression that you want to in the AP's if you follow these simple procedures:

A. Create any expression using the ***Expression Builder*** *shown in Figure 10-8.*

B. Click the ***OK*** *button.*

C. Click the ***Options*** *button, shown in Figure 10-7, to bring up the* ***Expression Builder****, Figure 10-8.*

D. Highlight everything in the expression box, and begin typing your own expression by hand. ***Do not use*** *the* ***Expression Builder****.*

E. Click the ***OK*** *button when finished.*

If you have typed a valid expression, even if it is not against the ***Contact1/Contact2*** *tables, which is all that the* ***Expression Builder*** *allows, your expression will be saved, and utilized.*

Note

A previous reader expressed concern that she did not realize that I was using two single quotes (') in my expression. She interpreted the two single quotes to be a single double quote ("). Hence, in this edition of the book at least, I have changed the expression to utilize the dBase alternative to ' or ":

dtos(Contact2.uDOB) > []

...the opposing square braces.

Note

Unlike the ***Filters****, which utilizes a similar dialog form to that shown in Figure 10-8, this Expression Builder* ***does not*** *require balanced dBase & SQL statements. A simple dBase expression will suffice.*

Note

Because I am looking for a birthday that occurs in a range of 5 days, and I would only want to add the ***Birthday Process*** *once during that range. I, therefore, employ a* ***Flag*** *field. The first* ***Sequential Event*** *of the* ***Birthday Process*** *will stuff this field with a character so that this* ***Filter*** *condition can not be met, hence, the* ***Birthday Process*** *will not be attached a second time during the period that the Flag is in place.*

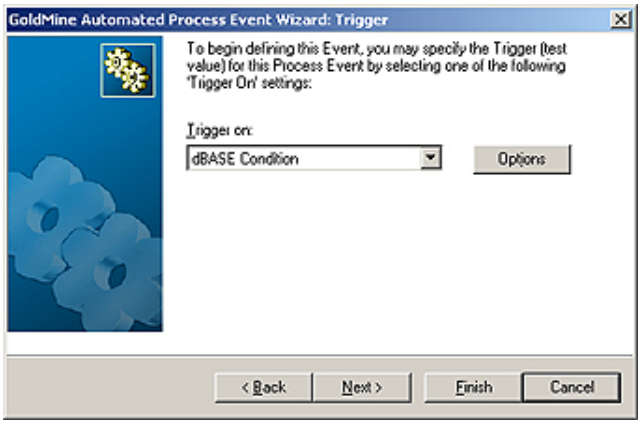


Figure 10-7

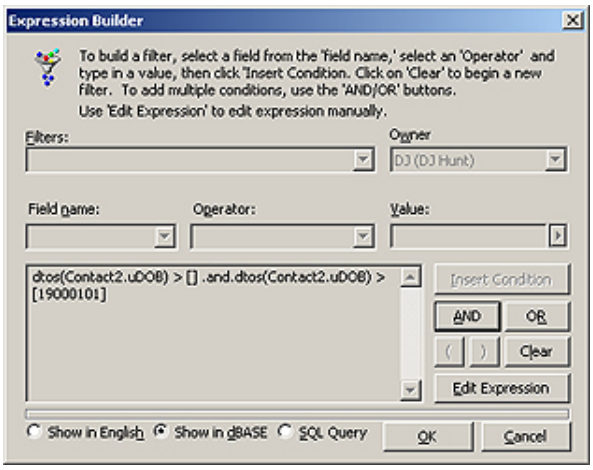


Figure 10-8

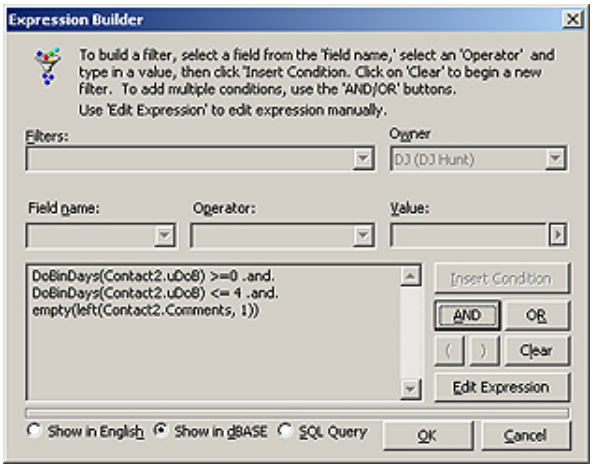


Figure 10-9

pressions by hand. Follow the sidebar Trick, and you should be okay for all future hand built expressions. In the expression, I am just testing two possible conditions to assure that the field contains a birth date. You must test for both cases, especially as your backend is SQL. A SQL date based field has a default string date of 19000000, hence, it is not empty, yet it is not a valid birth date. Although I don't actually have to test for an empty date field, I do. That is the old dBase programmer in me, and it doesn't hurt to test for empty in the off chance that the field is actually empty. Again, not likely, but there is always that chance. At this point I would click on the **OK** button to save this expression, and then click the **Next >** button to proceed along the Wizards path.

Heavy duty stuff isn't it? I'm getting a brain cramp just trying to write about this. Yet, we haven't seen the worst of the expressions.

In the next Wizard screen, no figure shown, I do select the ☑ **Trigger Filter** option. I then click on the **Filter** button to bring up that Expression Builder a second time. I follow the Trick discussed in the sidebar, and then enter my filter expression by hand as this expression could not have been created using the Expression Builder.

Look at the expression that I've employed in my trigger, Figure 10-9. I have employed the **DoBinDays()** function (Appendix A) to make sure that the birthday is greater than or equal to **0** days away, and less than or equal to **4** days away, and that our Flag field (see sidebar Note) Contact2.Comments, first byte, is empty. Notice that these are boolean **.and.** conditions such that all must return a **True** before the expression itself can be considered to be **True** or triggered. Having finished that, I click on the **OK** button, and then on the **Next >** button to bring up the next page of the Wizard.

If you are following along, you should select the **Perform Action:** of **Add a New Process** if this trigger is pulled. I have then clicked on the **Options** button to bring up the dialog form. From the dialog form I would ask you to select the, already established, **Birthday Process**, while checking the option to ☑ **Process immediately**. Now click **OK**, then **Next >**, and then **Finish** which concludes the building of this Event.

As I eluded to earlier, I am going to expand this chapter for this book, as many of you are still baffled by the complexities of the GoldMine Automated Processes, and rightly so. What I thought that I would do for this edition of the book was to show you how to utilize a single step Process to attach a historical activity en massé to a Filter/Group. This was so often requested that Richard Young of Prior Analytics Limited created a little applet to do just this, however, you know my philosophy:

Create a Filter/ Group Historical Activity

"**When it can be done within GoldMine, do not do it outside of GoldMine**".

So let's begin by creating a single step Process. Right-click at the top of the tree in the Automated Process Manager over the words **Automated Processes**, and select **New...** from the local menu. As you have already seen in the past, this begins the **GoldMine Automated Process Wizard**. In the **Process Name:** field enter **Create a Historical Activity**, and why don't we enter CHA into the **Process Code:** field, while leaving the default **Owner:** value alone. Now let's click on the **Next >** button. That's it, Step 1 of the wizard is finished.

On the screen of the wizard, we will only select 3 of the options:

- ☑ **Allow only one attachment of this process per contact**
- ☑ **Execute this process immediately upon attachment**
- ☑ **Allow users to attach this process**

All of the other options should be unselected, and now you may click on the **Next >** button for this step of the dialog form.

You should now be at the **Events** step of the dialog form, and it is here that you would like to click on the **New** button. I'm going to show you the completed dialog form here in Figure 10-10 so that you can see exactly what I've created.

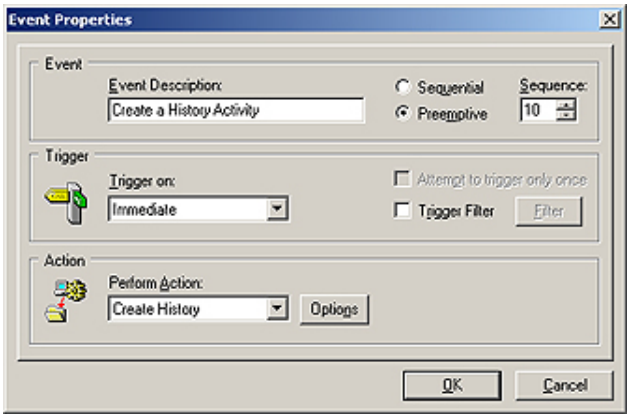


Figure 10-10

As you can see above, I have entered a **Event Description:** of **Create a History Activity**, while making this a **Preemptive** event with a **Sequence:** number of **10**. Naturally, I would want the **Trigger on:** to be **Immediate**, and, as you would expect, the **Perform Action:** is set to **Create History**. After all, isn't that what we are trying to achieve here?

Figure 10-11

Now I want you to click on the **Options** button. I will now be referring to Figure 10-11 shown here. You will notice, for the **Activity Type**, that I have selected **Others**. When you are creating this, you may select any RecType to your liking. For our **User**, I have selected to utilize the ◉ **Attaching user**. Again, you may not find this appropriate for your activity, and that decision remains with you at the time that you are modifying your Automated Process. To continue on, you would want to click upon the **Activity Details** button to bring up the **Complete an Other** dialog form as shown populated in Figure10-12.

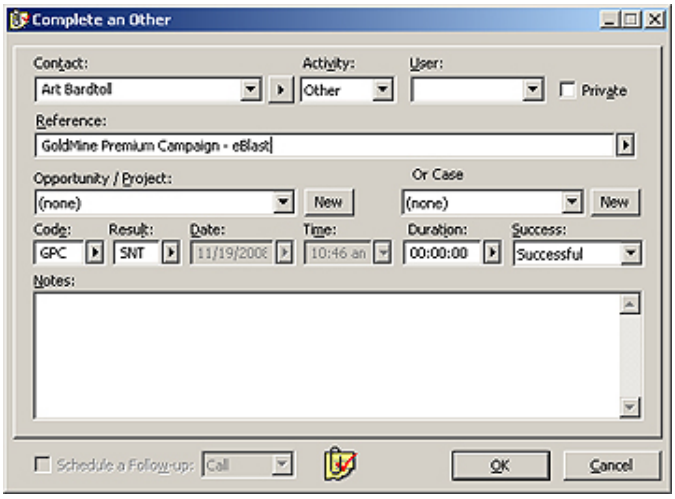


Figure 10-12

Many of the fields will be populated at runtime by GoldMine so I have only pre-populated the **Reference:**, the **Code:**, and the **Result:** fields. You could, if you want to, even include **Notes:** for this particular series of completed activities. When finished, click upon the **OK** button, and then upon the **OK** button of the **History Action** dialog form. Finally, you can click upon the **OK** button of the **Event Properties** dialog form, and then upon the **Next >** button of the **Events** Wizard. Yes, there is one more button, the **Finish** button, and then we are through with this stage.

Recommendation

*Yes I always use a **Code** as well as a **Result** value in all of my Completed activities, and I highly recommend that you do this as well.*

For Instance: You might make that code specific to your campaign.

Note

*You will retain this single **Event** Automated Process in your repertoire of tools, however, the dialog form shown in Figure 10-12 is the one that you will need to change each time you want to utilize this Automated Process to create a different series of historical events for a Filter/Group.*

Each time that you want to utilize this process to create a historical activity for a group of individuals you will first need to change the contents of the Events Completed Activity. Okay then, let's wrap this project up. We have our single Event process, so now what do we do with this? First I would ask you to **Activate** a small Filter/Group that you may have lying around. If you do not have one then simply **Tag** a couple of records, and that will suffice. With your Filter/Group activated select:

Tools
Automated Processes ►
Execute Process...

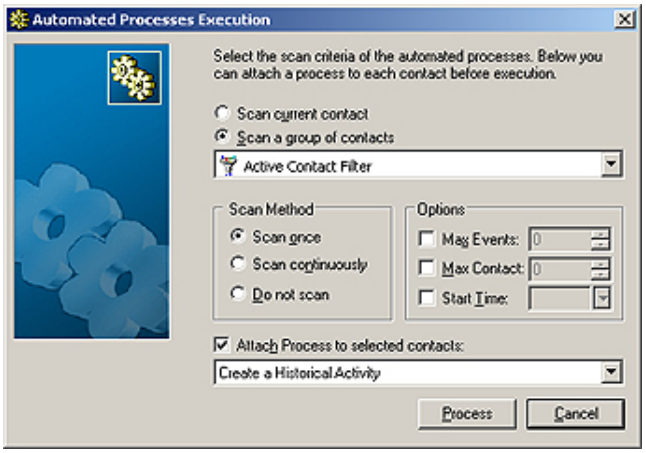


Figure 10-13

You will notice that the ◉ **Scan a group of contacts** is selected in Figure 10-13, and that the field will display **Active Contact Filter** by default when a Filter/Group is activated. What I have selected, that is not selected in the default state, is to ☑ **Attach Process to selected contacts:**, and with that I have chosen the process which we have just created called **Create a Historical Activity** using the drop-down menu. When you click on the **Process** button this Automated Process will be attached, to all of your Filtered/Grouped records, and because this process has been set to fire immediately upon attachment, the Event will process immediately, creating a new historical activity for each contact in the Filter/Group. Once the historical activity has been created, the Process will remove itself from the record as there is only a single Event in the process. When the **Process Monitor** indicates that you have finished with this activity, protocol dictates that you should release your Filter/Group.

For any future campaigns where you would like to repeat this action, you simply change the historical activity in the Event, and then attach that Process to the new Filter/Group. Could it get any simpler than that?

What I thought I would do now would be to look at each **Trigger** and, later, each **Action**, and I would discuss them in more detail. Although the Triggers have not changed in this edition of GoldMine Premium, the Actions list has had a new entity added, **Create Case**. Here I show you all of the possible Triggers/Actions for this edition of GoldMine Premium:

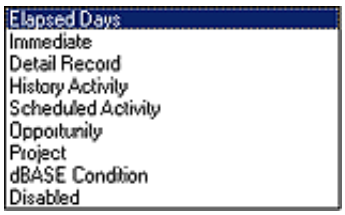


Figure 10-14a

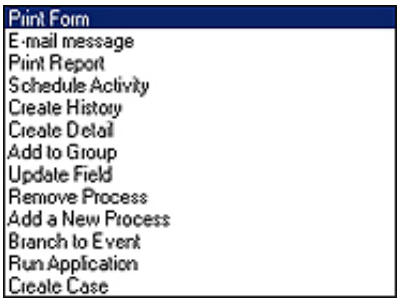


Figure 10-14b

Triggers

Elapsed Days: This trigger can be a bit tricky. I have seen users believe that this is the elapsed days since the process was added when, in fact, it is x number of elapsed days since the last Sequential Event had been triggered. You would probably never use this trigger in a Preemptive Event.

Immediate: This trigger is self evident. It is how one would turn a Sequential Event into a single use Preemptive Event, so to speak. You could use this trigger in your Preemptive or Sequential Events, however, in a Preemptive Event this would trigger every time that your processes are scanned, and you may want that to be the case yet I would doubt it. In my opinion, this is not a condition that you are likely to ever want to use in a Preemptive Event. That being said, I have used this when populating fields with default values, when there were too many default field values for me to employee the Lookup.ini for my client.

Detail Record: As can be seen in Figure 10-15, it is possible to trigger on ◉ **Details**, but you could trigger on other ContSupp RecTypes (yes the RecTypes field is what GoldMine uses to differentiate

between the different ContSupp records as you read previously in The Tables chapter). You can also trigger off of:

- Document Links
- Additional Contact
- Referrals

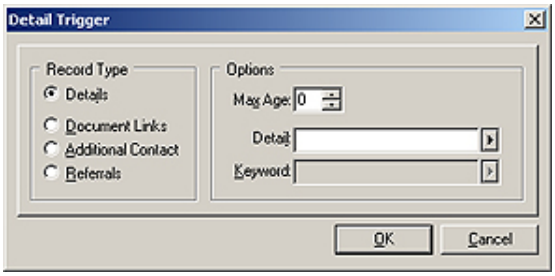


Figure 10-15

That is, you can trigger off of the creation of one of these RecTypes into your ContSupp table. Therefore, you must also tell the trigger properties how far back you are willing to look for the creation of one of these RecTypes from the day upon which the Event is being scanned.

Note

*Setting **Max Age:** to **0** days means if that Contact record has that RecType existing on any of its ContSupp records, regardless of the day that they were created, well that would be enough to trigger this Event. The 0 value is rarely used except in a sequential event, and never used in a preemptive event without the additional use of a **Flag** in the **Trigger Filter**.*

Think about this now. If you make the **Max Age:** 5 days, and you scan your processes daily, then this trigger would be pulled every day for the next 5 consecutive days following the creation of this RecType if this were the trigger for a Preemptive Event. Not, probably, what you were trying to achieve, hence, I refer you back to the use of Flags in the Trigger Filter to prevent this type of Event from executing more than once. On the other hand, if you enter a Max Age: of 1 day, and the RecType is added on Friday, and you don't scan your processes until Tuesday next, because of a Monday holiday, you will not trigger this Event either. What a conundrum. Personally, I would rather use the **Max Age:** of 5 days with a Trigger Filter on a Flag field for my Preemptive Events than to risk the chance of not pulling the trigger on this Event at all. Naturally, this is a moot point for Sequential Events as a Trigger can only be pulled once per Sequential Event before the process moves forward to the next Sequential Event.

You can further refine your criterion, when ◉ **Details** is selected, by picking a **Detail:** from the F2 Lookup List such that you are only looking at the new **P** RecTypes that contain that value in the **Contact** field (you see I know this because I've read The Tables chapter in this book). Also, you can further refine your trigger by using a **Keyword:**. This field is available for refining your trigger regardless of the RecType that you wish to trigger against, whereas, the **Detail:** field is only available when the ◉ **Details** radio button option is selected.

History Activity: As can be seen in Figure 10-16, this type of trigger is similar to the Details trigger. You first have a frame for the **Activity Type** (aka RecTypes), with a frame for **Options**, and a frame that I have never had the opportunity to utilize labeled **Outcome**.

Figure 10-16

In the Activity Type listing, you have most of the possible RecTypes that the ContHist table could possibly contain. Refer to The Tables chapter for the various RecTypes that are utilized here. You can only trigger on a specific ContHist. RecType, and there is no option for Any History Activity.

Note

*The default, **Max Age: 0**, tells GoldMine that any History record of the specified RecType that was ever created against this Contact record is satisfactory to Trigger this Event.*

Be weary of leaving the default setting unless you have an absolutely good understanding of what, and why you are doing so.

You would, most likely, not want to utilize this setting in a Preemptive Event.

I don't think that I need to rehash **Max Age: 0** day for you, however, you are being introduced to three new trigger refinements, **Activity:** (ContHist.ActvCode), **Result:** (ContHist.ResultCode), and **User:** (ContHist.UserID). These three should be self explanatory. And, even though the **Ref.:** field is a different nomenclature, it acts, and reacts similar to the previously discussed **Details Keyword:** field.

Although a History trigger allows you to refine further for ◉ **Any Outcome**, a ○ **Successful**, outcome or an ○ **Unsuccessful** outcome, in my 20 years of working with GoldMine, I have never seen anyone use this flag when Completing an activity within GoldMine. That being said, it has been a useless refinement to the triggers for my clients, however, that does not mean that your organization may not want to use this trigger refinement.

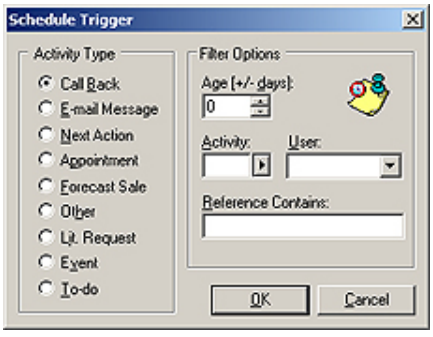


Figure 10-17

Scheduled Actvity: This brings us to the Scheduled Activity Trigger, and, as with the History Activity, we can see that the Trigger dialog form is pretty much the same, Figure 10-17. Naturally there are fewer

RecTypes to use as triggers, and, as there can be no outcome with a scheduled activity, there is no outcome refinement area. This holds true also for the fact that you see no **Result:** field on this trigger dialog form.

Having said all of that, you may have noticed that there is yet another difference. Instead of **Max Age:**, you will notice **Age [+/- days]:**. As this may be throwing a curve ball at you, I think that I will examine it in a little more detail for your benefit. Normally, one would not schedule an activity for a past date. As a test, I scheduled a Call for tomorrow. I set the trigger Age [+/- days]: to +3 days. I attached the process which was set to scan immediately upon attachment, and absolutely nothing happened. I then changed the date of the Pending activity to yesterday, and scanned the Process which did Trigger on this pass. To further test this out I changed the Age to -3 days, and I changed the Pending activity OnDate ahead to tomorrow again. I scanned the Process again, and this time it Triggered again, and it Triggered every scan as long as the OnDate was 3 or less days into the future. When I moved the OnDate to 4 days into the future and did the scan the Trigger was not pulled. Also, and just to complete my testing, I set the OnDate to yesterday, and when the age was set at -3 days, and the AP's where scanned this Trigger was again not pulled.

Here is what the GoldMine Help files have to say about this field:

> ***Age (+/- days):*** *Specify, in days, how recently the Scheduled Activity must have been scheduled on the calendar. Use the plus sign (+) to indicate past days, and the minus sign (-) to indicate upcoming days. Example: if you specify +3, it triggers the event only if a calendar activity was scheduled for completion within the last 3 days. If you specify -3, it triggers the event only if a calendar activity is scheduled for completion within the next 3 days. The default value of 0 = an activity on any date triggers the event.*

Well, that certainly seems contrary to my way of thinking. A **+** equals the past, while a **-** equals future. I guess that I am from the old school, and that I do not understand this logic. In my school days, don't ask when, you went back in time (**-**), and you went forward (**+**) to the future. However, as long as you understand the developers logic here, then you should be able to create a working Trigger to meet your constraints.

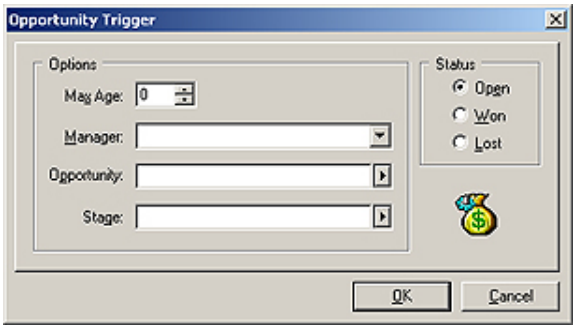


Figure 10-18

The next Trigger that I want to examine is an **Opportunity Trigger**. Clicking the **Options** button on this Trigger brings up the dialog form shown here in Figure 10-18. In the **Max Age:** field, you would want to select the age of the Opportunity in days. You may use the spinner to increment or decrement this number, however, I am not sure why you would decrement the number past 0, although you can.

You could cull your requirements down further by selecting a UserID for the Opportunity from the drop list associated with the **Managers:** field. You might also want to limit Opportunities that were of a specific type (name) in the **Opportunity:** field. In the Options frame there is one more filterable option, and that is for the **Stage:** of the Opportunity. Lastly, you see a radio button frame, **Status**, where you may further limit your criterion to either an ◉ **Open**, ○ **Won**, or ○ **Lost** Status for the Opportunity.

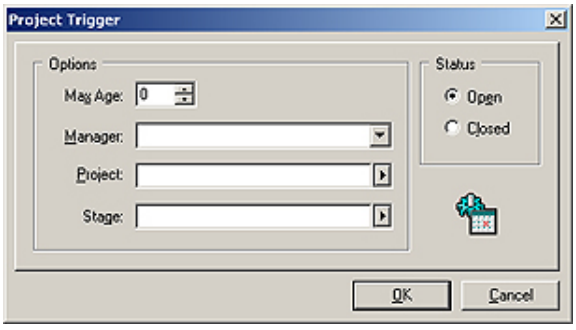


Figure 10-19

There is very little difference between the Opportunity trigger, and the **Project** trigger as you can see here in Figure 10-19. In the **Max Age:** field, you would want to select the age of the Project in days. You may use the spinner to increment or decrement this number. Again, you have the **Manger:** field, this time the **Project:** field, and again, the **Stage:** field to further refine your trigger criterion. This time, in the **Status** frame, you only have the two possible radio choices of either: ◉ **Open** or ○ **Closed**. You understand, that either a Project is on going or it has been completed, hence, only a need for one of two options. Pretty much, everything that applied to the **Opportunity** trigger is applicable to the **Projects** trigger.

Note

*The default, **Age [+/- days]: 0**, tells GoldMine that any Scheduled activity record of the specified RecType that was ever created against this Contact record is satisfactory to Trigger this Event.*

*As I'm interpreting this, a **+** indicates Pending activities scheduled for a previous date, while a **-** indicates Pending activities scheduled in the future.*

Note

*My testing has shown me that a **-3 day Trigger** option will be pulled for any **Cal.OnDate** that is within 3 days into the future from the date of the scan.*

I say this, not so much because it is an issue, as it is just a functionality that you should be aware of when creating your Events.

Note

*I believe that the **Disabled** trigger was intended for developers use to turn off certain known functioning events so as to not have to process them each, and every test scan. This is purely speculation, however, I can think of no other use for this trigger.*

The next trigger, **dBASE Condition**, I have already covered in this chapter a few times, and do not choose to elaborate any further on it in this section. However, the last trigger, **Disabled**, interests me in as much as I have no clue if it even functions, and if it does function, what exactly is its function. I will investigate that now for you.

The GoldMine Help file for GoldMine Premium states:

"Turns off the trigger condition setting for the selected event."

But I would have to ask myself, Why? Is this so, you can leave the Event in the process, but not have it active? If this is the last sequential event will it be tested, and the process be removed? Give me a few minutes. I want to go run some tests.

TaDa! Testing has shown me that the Trigger is not tested, and that the Event is ignored. If this is the last Sequential Event, then the Process is removed. If this is not the last Sequential Event, this Process is just by-passed, and GoldMine moves directly to the next Sequential Event. I'm baffled. This trigger works exactly as defined in the Help dialog. What is this world coming to?

Actions

Let's move on. As that was the last trigger, I can now move my discussion forward to the various **Actions**. First, and foremost, you must understand that the Event Actions can do no more than you could do within GoldMine, and sometimes less because they have to function automatically. There is no user logic in the Automated Processes area that will make human decisions for you.

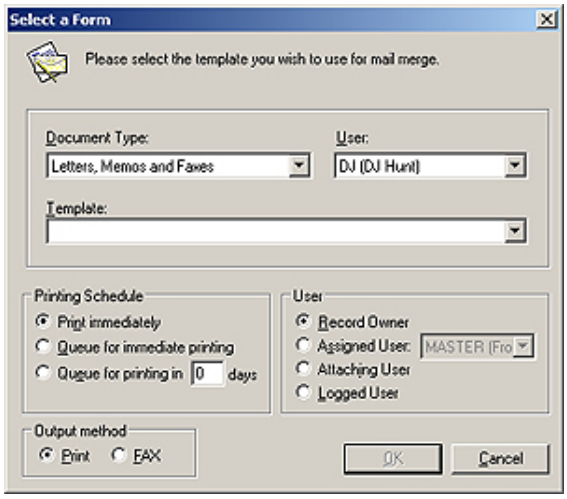


Figure 10-20

The first **Action** on the list is **Print Form**, and, quite simply, this allows you to print a templated form. First off, let me state that I never, ever use this option in an Automated Process. First: this is an unattended and automated process, and you never know if there is paper in the printer, if the printer is on line, if the printer ink has run out, etc. I suppose, if you had the next Event trigger immediately, and it was set to schedule someone to check to see if the print job occurred, and if so to handle it, then it may be an acceptable Action. It is my thinking, however, that you should not even bother with this Event, and only have a single Event to automatically schedule someone to do the print job, as well as the handling of it after the printing. That would be the extent of automation that I would want in this process.

Having said that, you can see in Figure 10-20 that you can select any **Document Type:** for any **User:** using any of their templates, **Template:**. Once selected, you may choose a scheduling method in the **Printing Schedule** frame. You have the default ◉ **Print immediately**, or ○ **Queue for immediate printing**, and finally, ○ **Queue for printing in XX days**. The queue concept is the most difficult to understand so I ran some tests so that I could better explain it to you.

For instance: looking at Figure 10-20, if I had selected ◉ **Queue for immediate printing**, and as I also selected ◉ **Assigned User: Master**, I then ran the event, and this is what my debug log showed:

```
0[47] Automated Processes  [10:51 am - 11/29/2008]
0[47] Scanning Contact: Computerese; DJ Hunt
0[47] Read Process:  Test for Book  [Next Event: 100, Process: BYCSUEV${:] R#Y]
0[47]   100   Calendar +/-
0[47]   110   Second Event
0[47]         --> Triggered.
0[47]             Queue Form  [3MJ2QBC.XN${/72]
0[47]    Remove ended process: BYCSUEV${:] R#Y
4[47] Automated Processes: 1 Contacts; 1 Scanned; 1 Triggered; 1 Pass(es)  [Ended 10:51 am -
      10/29/2008; Dur: 0:00]
```

However, and this is where everyone gets lost, "*Where the heck was it queued?*", and "*How do the queued print jobs get automatically printed?*". This is where we have to look at two, yes two other areas of GoldMine. To see where the job gets queued, you need to look into the GoldMine **Literature** tab, and, in this case, under the **User: Master**. You should be able to find the job under the tree branch **Queued Documents**. You could right click on the document, and select **Fulfill** from the local menu, however, where would the automatic in that be?

Note

*In this edtion of GoldMine Premium **Literature Fulfillment** has been truncated to simply **Literature**.*

Location:

GoTo
Literature

This brings us to the "*How do the queued print jobs get automatically printed?*" question. For this answer, we have to look into the GoldMine **Server Agents**, under the **Print/Fax** tab. Yes, you have to set up Server Agents, and, as with GoldMine Automated Processes, GoldMine must be running, and someone or something must **Start Server Agents**. Ah ha! There goes automatic right out the window.

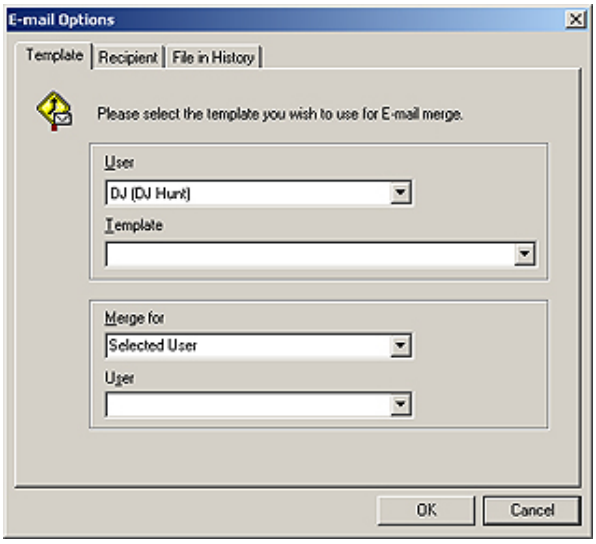


Figure 10-21a

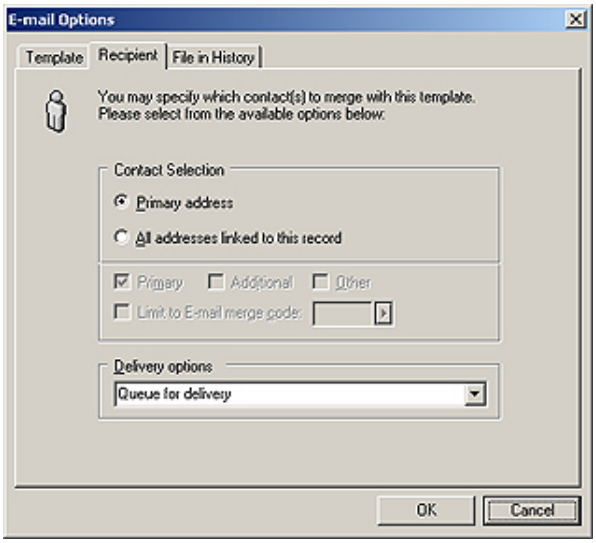


Figure 10-21b

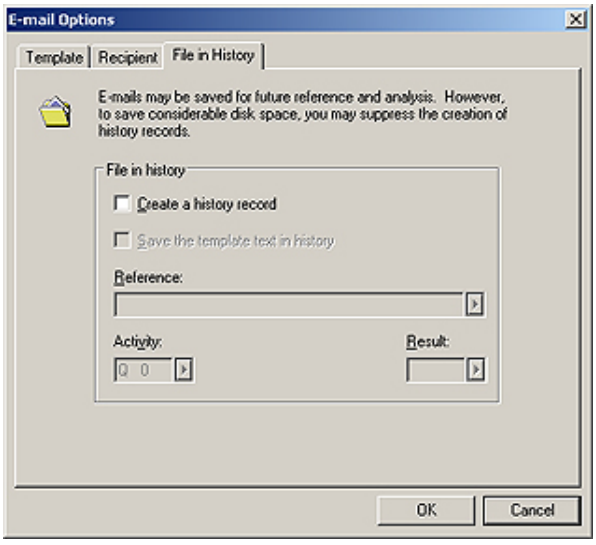


Figure 10-21c

Note

*Should you choose the **Merge for** option of **Selected User**, GoldMine will only permit you to select a UserID for which there is a defined E-mail Account under that UserID's Options.*

Note

*Selecting the ☐ **Other** option does not permit you to select another E-mail Address for the primary contact. My assumption here is that your GoldMine will automatically choose the next E-mail Address for the Primary Contact that has not been marked as the Primary E-mail Address, yet which is associated with the Primary Contact. The GoldMine Help files were not very conclusive on this definition.*

***Other:** Sends to a different e-mail address of the primary contact.*

Note

*If this is an Automated Process, and it is, you should think hard an strong as to whether you really need to ☐ **Save the template text in history**. Personally, I don't believe that this is necessary.*

*Also, don't ask me why the default value for the **Result:** field is a **0**. I have no clue. It was probably just the result of some buggy coding in that area.*

Okay, let me wrap up this Action which, as I stated previously, I never use anyway. You can see that there is a frame called **Output method** from which you can choose to ◉ **Print** or to ○ **Fax**, and that there is another frame called **User** from where you may select the user to be the ◉ **Record Owner**, ○ **Assigned User:**, ○ **Attaching User**, or the ○ **Logged User**. I believe that all of these are self explanatory requiring no further explanation on my part.

And, moving on to the next Action of **E-mail message**, and click on the **Options** button for this action will make available the dialog form shown here in Figure 10-21a. As you can see there is a lot that has to be defined for this type of action for it to be utilized properly.

On this dialog form, you must select the template to be used, and, as you know, each user could have their own set of templates. The Action doesn't care whose template you use, but you must specify the **User:** as well which **Template** of that UserID to be utilize. You can then tell GoldMine from whom the merge should appear to be coming. You will have 4 choices (not shown) under the **Merge for** option, and they are:

Record Owner
Selected User
User who attaches this Process
User who scans this Process

The **User** field will only be active if you select to merge for the selected user. If you have chosen this option, you must select a UserID from the supplied list for this field. GoldMine will then use the Option settings, as defined for this UserID, when sending an E-mail via this particular Action.

In Figure 10-21b, you can see that you must define some **Recipient** information. Whether this e-mail is to go to the ◉ **Primary address** only, or whether to send it to ○ **All addresses linked to this record**. The next grouping is inactive, and remains so, unless you choose ○ **All addresses linked to this record**. Selecting this will allow you to further clarify if you want this e-mail sent to the ☑ **Primary**, ☐ **Ad-**

Note

*In this build of GoldMine Premium on the **File in History** dialog form, you will notice that there are two Hot Keys that are the same:*

Reference:
Result:

I have reported this to FrontRange, and I expect that it will be fixed before the build goes Gold.

ditional contacts, or ☐ **Other** E-mail Addresses for the primary contact. You may further ☐ **Limit to E-mail merge code:** from where you can enter, or select from your F2 Lookup list, a merge code. If you decide to use this option, you should consider using the F2 Lookup list of Merge Codes.

Lastly, you have your **Delivery options** (although plural, you may only select one method for transport):

Queue for delivery
Send immdiately
Save as draft

Figure 10-21c, **File in History**, is the last dialog tab form. Here you determine if you want the Event Action to ☑ **Create a history record** for you. Once selected, all of the other options on this tab become enabled. You may now ☐ **Save the template text in history**, and you may override the **Reference:** with your own information instead of using the template subject in this field. As well, you may select, or add, your own **Activity:** and **Result:** codes which I highly recommend if you plan to use this feature.

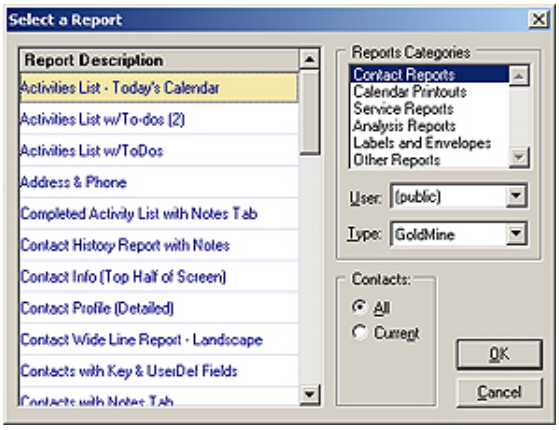


Figure 10-22

This, then, concludes the E-mail message action for your events, and allows us to move forward to the action of **Print Reports**. First, and foremost, let me state that I hold the same reservations against the use of this action as I did for using the **Print Form** action. The printer must be on line, someone must know that the report was supposed to be there, etcetera, etcetera. Again, my first choice would be to schedule someone to perform this action, and to not choose this Action ever at all. With that in mind, you are offered this dialog form from which you may select the report (any report, belonging to any UserID within GoldMine). You must also determine if the report is to be printed against ◉ **All** contacts, or only just the ○ **Current** contact. Previously you had the option of using a ○ **Filter or Group** of records, however, this option has been removed with this release of GoldMine Premium. You cannot predefine the printer to which the report is to be printed. This is pretty much another WYSIWYG, and there is very little else that you can do with this Action. Please consider my suggestion of not utilizing this Action, but instead, using the next Action that I will be discussing with you, the **Schedule Activity** Action.

So it comes as no surprise that the next Action that we encounter is the **Schedule Activity** Action. Well now, that seems like something that I would like to know more about as I've already mentioned it a couple of times in lieu of directly doing other Actions. Here you have it, anything that you can schedule by hand within GoldMine can be scheduled via an Event Action.

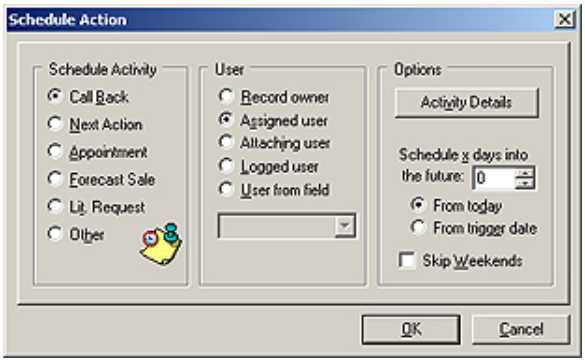


Figure 10-23a

Look at Figure 10-23a which is a result of clicking on the **Options** button, and Figure 10-23b, on the next page, which is a result of clicking on the **Activity Details** button in Figure 10-23a. I will describe Figure 10-23a first before moving on to Figure 10-23b. You know, the natural order of progression, a before b. Ah, let's just move on.

In the **Schedule Activity** frame, you have radio buttons, each representing a different RecType in the GoldMine Calendar (refer to The Tables chapter of this book). As with all radio buttons, you may only choose one of the available RecTypes.

The next frame allows you to identify the **User**. The first, and default option is ◉ **Record owner**. If you choose this option, and you have a single UserID identified as the Owner of a record, then the UserID from the Owner field will be used when scheduling an activity. Should you choose to select the option of ○ **Assigned user**, then the UserID that you have specified under **Activity Details**, discussed later, will be employed. Then we have the third option, ○ **Attaching user**, which I have

Tip

*Should you follow my lead on this one, you would be wise to modify your **Lookup.ini** to auto-populate these fields when a new record is created in GoldMine. Remember that GoldMine cannot schedule an activity if there is no **UserID** in a field. I use the Lookup.ini as you can only use fields from the **Contact1/Contact2** tables for this activity, and, coincidently, the Lookup.ini can only populate fields in the Contact1/Contact2 tables.*

never had the opportunity to use, however, this applies to the UserID of the person who will be attaching the process to the record in the first place. Holding the fourth position is ○ **Logged user**, which is another option that I have never had the opportunity to employ, however, the obvious here is that this means that the UserID of the Workstation that is currently scanning the processes will be used when scheduling an activity. Which brings up our final contender, and the one that I prefer to use most often, ○ **User from field**. Yes, I make it a habit of assigning each record a series of fields, Sales Representative, Account Manager, Manager of Account Manager, etcetera, and I always select this radio button when scheduling an activity. I then proceed to point to the appropriate field that will contain the UserID for this or any scheduling Event.

I'm going to pass on the **Activity Details** button for the moment, and finish with the remainder of the **Options** frame, **Schedule x days into the future: 0**. Well this is almost obvious, however, when you throw in the radio selections this becomes a little confusing, at least to me it does. You have the choice of selecting one of two options, ◉ **From today** or ○ **From trigger date**. Choosing the second radio option produces a crystal clear consequence, the activity is schedule x days after the trigger date, on the other hand, selecting the default option is not quite as clear. From the GoldMine Help files:

> **Select one:**
>
> **From Today:** Starts counting days from the current date.
>
> **From Trigger Date:** Starts counting days from the date the event is triggered.

Well then, you have to ask yourself, exactly what is the current date? The date that I'm developing the Event? The date that I attach the process? The date the process is triggered (I think not or why have the second option)? I really can't figure this one out, and obviously, the GoldMine Help files were, well, or no help at all. Please let me know if you've figured this one out.

Next is a single checkbox option which is self descriptive, ☐ **Skip Weekends**.

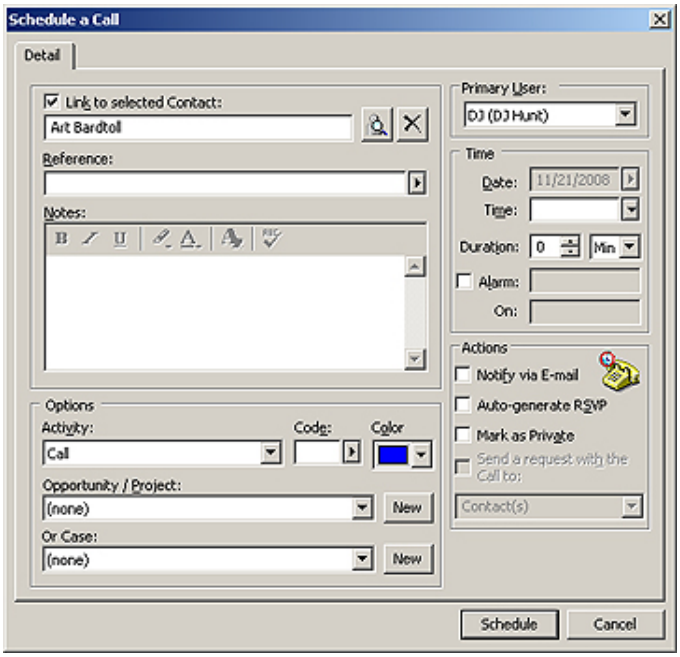


Figure 10-23b

This now brings us full circle, and back to the **Activity Details** button, refer to Figures 10-23a & 23b. It is here that we may define some specifics of the activity to be scheduled. This is a normal **Schedule a Call** dialog form except that some of the fields have been disabled because we are Scheduling an Activity via an Event Action.

First, let me say, that even though you can select a different contact, don't do it as it is a wasted effort. This field is populated automatically as a result of the triggering of this Event, and the Action defined. You may, however, choose to accept the default of having the Action scheduled call ☑ **Link to selected Contact:**, in fact, I see no reason to change this from the default at all.

In the **Reference:** field you could get a little creative if you so choose by utilize dBase syntax functionality. For Instance:

> **[Call]+&FirstName+[to follow up on order]**

You are reminded that the **Reference:** field does have a character limitation, but within that limitation, you may use any valid dBase expression.

Notes: would probably not be a good option for an Automated Process, however, you could add standardized notes should you so desire.

Utilizing an Activity Code, **Code:**, is always a recommendation from my mouth. Personally, I use AP so as to designate that the activity was scheduled by the GoldMine Automated Processes, and not by an individual, however, this will only work properly if you have **Force valid input** turned off for this Codes F2 Lookup List.

Personally, I would not select an **Opportunity / Project:** for an Automated Process entry, yet that is strictly up to you as to whether you would utilize this or not.

At this point, you would want to define your **Primary User:** if you had selected ○ **Assigned user** when on the previous dialog form, otherwise, just leave this field alone.

You may designate a **Time:**, however, as this is an Automated Process, you could get a lot of activities scheduled via the AP for the same user, on the same day, at the same time, for the same duration. Personally, I enter a character here, **A**, forcing this activity to the **Task** (untimed) area of the calendar as an untimed activity. Your end users will be much happier with this resulting activity than they would have been with the default timed activity.

Lastly, I always select the ☐ **Alarm:** checkbox. Did I say always? Yes siree, I certainly did. I prefer the in your face approach on activities that are scheduled automatically for a user. Do you really believe that your end users will always look at their calendars? I think not unless you somehow tie their paycheck to the calendar. However, they can't miss an alarm that pops up in their face each morning when they log into GoldMine.

I have never selected anything from the **Actions** frame, although it is certainly possible. I won't go through the three options that you have in this frame as they are all self descriptive. If you do utilize these in your normal schedule paradigm, then you may want to consider employing them via your AP's as well. For instance: ☐ **Mark as Private** may be something that your organization employees for scheduled activities. This may be a Security paradigm that your organization chooses to utilize.

Note

This type of Action is often used in GoldMine where no history was created when doing an eBlast for a campaign.

People would create a single Event process that would be processed immediately upon attachment, that would trigger immediately, and would create a history record to identify this record as having been a member of this eBlast. Attaching this Process to the Filtered group would then create a history activity for each member of the Filter group.

*Refer back to the section of this chapter titled: **Create a Filter/Group Historical Activity**.*

In list order, we now have another Action to look at, **Create History**. This is another two dialog form Action as you would expect because what is a History activity after all, but the completion of a Scheduled Activity, hence, one would expect these two Actions to be similar in nature. As an Automated Process cannot complete a scheduled activity, the Actions of the AP Events permit you to create a history activity. In point of fact, I utilized this very Action earlier in this chapter with you in the **Create a Filter/Group Historical Activity** example.

Figure 10-24a

Figure 10-24a is the first of the two dialog forms that one will run across. Again, GoldMine has the **Activity Type** frame that this history record will be associated to, and, again, the list conforms to those RecTypes that are available within GoldMine. There is no difference in the **User** frame selection options from those that I have previously mentioned for a Scheduling Action. Refer back to that if you need a refresher on these options. Again, we have our **Activity Details** button (with a different Hot Key this time), however, this time GoldMine produces a variant of the **Complete a Phone Call** (if you had selected the RecType of ◉ **Phone Calls**) dialog form, Figure 10-24b. There is not much more that I can explain about the filling in of this form, however, I would remind you that you could get inventive with your **Reference:** field information. Additionally, I always recommend a **Code:**, and a **Result:** be input.

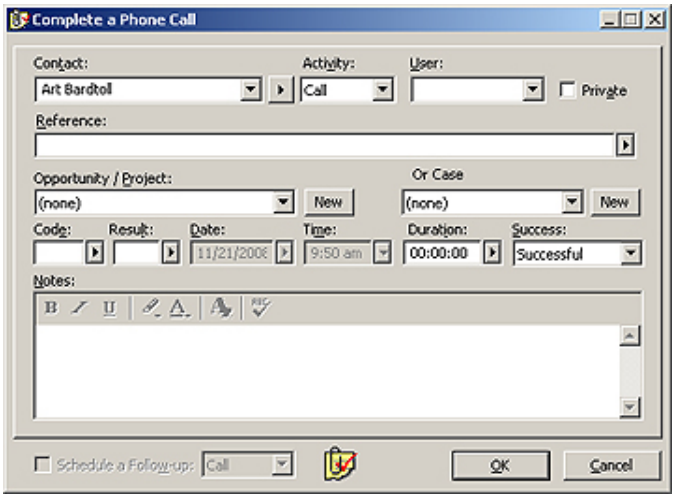


Figure 10-24b

Figures 10-25a & 10-25b show us the Action for **Create Detail**. You are allowed to select from any of your pre defined Details, and/or create a New Detail. If you have set up the **Info** tab (and you can't

Note

The extended Details have now exposed 12 fields:

Field 1: [35 characters, Title]
Field 2: [20 characters, LinkAcct]
Field 3: [20 characters, Country]
Field 4: [20 characters, Dear]
Field 5: [20 characters, Fax]
Field 6: [10 characters, Zip]
Field 7: [6 characters, Ext]
Field 8: [20 characters, State]
Field 9: [20 characters, MergeCode]
Field 10: [40 characters, Address1]
Field 11: [40 characters, Address2]
Field 12: [40 characters, Address3]

Note

As you cannot create Groups via the Actions of an Event, you must select a ***User:*** *that has Groups available. You will then be afforded the opportunity of selecting one of that users' Groups from the* ***Group:*** *drop list.*

You may have noticed that the ***OK*** *button is disabled. GoldMine will not even allow you to accept this Action until you have select a* ***Group:****.*

Note

The ○ ***Look-up a replacement value in lookup.ini*** *will only activate if you choose a field that appears on the left hand side of the equation under the AutoUpdate section of your Lookup.ini. i.e.*

[AutoUpdate]
Key1 = Key3

This works counter intuitive to the norm. If you actually wanted to update the Key1 field with the instruction set in the Lookup.ini for the Key1 field, you would need to have something similar to this in your AutoUpdate section:

[AutoUpdate]
Key1 = Key1

This basically says, if the Key1 field changes, then to apply the instruction set for Key1 to control how the field changes.

You can read more on this in the chapter which covers the ***Lookup.ini****.*

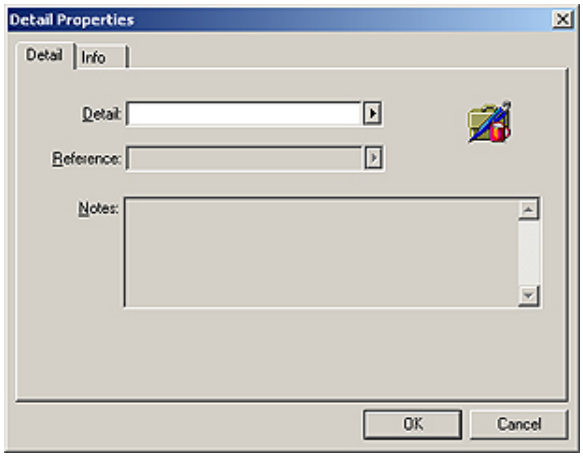


Figure 10-25a

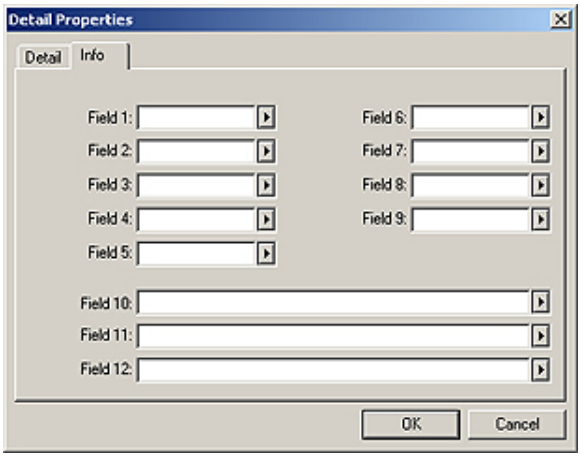


Figure 10-25b

Figure 10-26

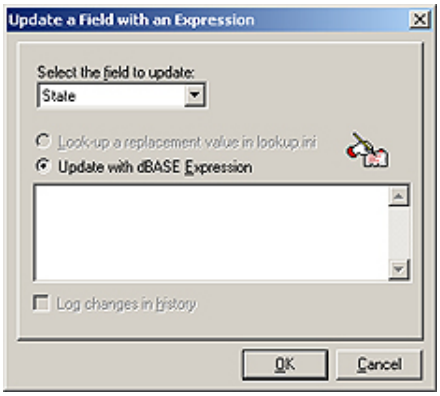


Figure 10-27

gonna happen.

do that here) then you would be allowed to fill in any of these exposed extended fields as well. There really isn't anything more that I can explain when it comes to creating a Detail via an Action.

You may have noticed, in Figure 10-25b, that GoldMine Premium now has exposed 12 fields from the ContSupp table for Details as opposed to the previous 8 fields in the GoldMine Premium 8.0 series. See the sidebar for the fields that can be utilized for a new Detail record in todays GoldMine Premium.

The Action of **Add to Group** is even simpler. There is but a single dialog form to complete as shown on this page in Figure 10-26. If the selected **User:** has any Groups available, they will be listed in the **Group:** field (you may not create Groups here). You may then add a **Membership Reference:**, or select one from the F2 Lookup list, and select a **Sort field:** from the drop list. Once you have selected a **Group:** the **OK** button will be enabled for you to accept this Action.

Your next available Action, from the drop list, would be the **Update Field** option. I use this one very often, especially when I have the need to use Flags in my AP's to prevent the same action from occurring more than once during a given period. You will notice, Figure 10-27, that I have selected the **State** field from the drop list associated with the **Select the field to update:**, so that the Lookup radio option is enabled (see sidebar Note). I'm keeping this in here as it functioned properly in previous builds of GoldMine Premium, however, in the build that I am currently using, and as you can see in Figure 10-27, this feature is disabled despite the fact that I have the proper section in my Lookup.ini.

So now, instead of entering an ◉ **Update with dBASE Expression**, you can simply select the ○ **Look-up a replacement value in lookup.ini** option, and let your Lookup.ini perform the appropriate task for you.

Barring that, you will have to create your own dBase expression, and remember, it is asking for an expression, and not for a value. For Instance: the Key1 field is a character field, and if we wanted to update this field with **Client**, we would need to enter **"Client"** or **[Client]** in the edit box. This edit box accepts any valid dBase expression, and will throw an error if you click on the **OK** button when there is an invalid dBase expression in the edit box. One reminder, always keep in mind the field data type that you are working with, and field characteristics of the field that you are updating. I mean don't try to stuff 50 characters into a 20 character field as it just ain't

Another new option, that has been added to this version of GoldMine Premium for this Action, is the ability to ☐ **Log changes in History**. If the field selected to be updated has its Security feature set to Log changes to History, then this option, as shown in Figure 10-27 on the previous page, will be enabled for selection. Sometimes you may not want a field that is updated via an AP Event to create a History record, and then again, sometimes you will. Hence, FrontRange has added this as an option for which you are capable of making the final determination.

That's enough said on the Update Field Action. Let's look at the next Action in the drop list, **Remove Process**. The first thing that you may have noticed, no figure supplied, is that the **Options** button is disabled. That's right, this Action does one thing, and one thing only. It removes this process, and only this process from the triggering Contact record. This Action cannot be set up to remove any other process other than itself, and that's all that I have to say on the Remove Process Action.

Tip

*The **Attach an Automated Process** dialog form is brought up in **Code** order. Every UserID that has an Automated Process has their process displayed here. If you are trying to find a process that DJ created, it is extremely difficult.*

*Assign a single letter or single digit as the 1st character of the **Code** field representing a single UserID in your organization. That will give you 62 different groups of processes that you can have, and you will be able to find those created by **DJ** because you know that all of his processes begin with a lower case **d**.*

Well, if we can remove a Process, you'd think that we could add a Process as well, and you'd be 100% correct. The next Action is to do just that, **Add a New Process**. Notice in Figure 10-28, produced by clicking on the **Options** button, that every Process that was ever created is brought up in the **Attach an Automated Process** dialog form (see sidebar Tip).

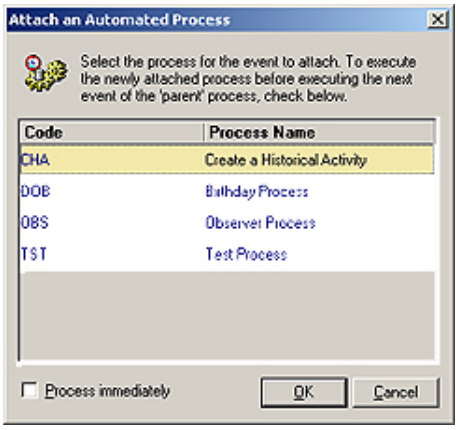


Figure 10-28

So, you see that you can add any Process as your Event Action that was ever created under any UserID as long as the process still exists within your GoldMine database, specifically the **Tracks** table. That's right. Although FrontRange may have changed Tracks to Processes on the dialog forms, the Table names have not been changed. Please notice the ☐ **Process immediately** option at the bottom of the dialog form. You must select this option if you want the selected Process to be processed immediately at the time that it is attached. As well, the Process selected must have been set to ☐ **Execute this process immediately upon attachment** (refer Figure 10-1). Other than that, there is nothing spectacular to say about the Action of attaching an Automated Process. So let's move forward.

Now we come to one of the biggies, **Branch to Event**, see Figure 10-29. Here you can get into some programming capability called branching script. Basically, this Action will permit you to branch to any other Event within this Process. Let me restate that, "*...any other Event within the same Process.*". You may not step outside the confines of the present Process using the Branch to Event Action.

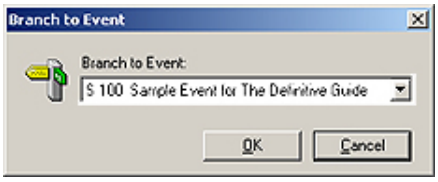


Figure 10-29

Tip

*Even the best of programmers occasionally programs themselves into an infinite loop. To assist in helping you avoid this situation, you should always **Branch to Events** that are Sequential Events, and that follow the Triggering Event.*

You must be very careful with your branching within the Process. You see, many times, a programmer will inadvertently program themselves into an infinite loop. This in itself is not bad. It is when they then turn around, and claim that the AP's don't work properly. Be very careful that you do not make this mistake. Adhere to the Tip in the sidebar, and follow the path of your branches to their logical conclusion to assure that you haven't placed them into an infinite loop. If you do happen to find yourself in this situation, you will need to shut down GoldMine to break out of the loop. Don't be embarrassed as you won't be the first to have done this, and I assure you that you won't be the last to do this. Where do you think the term, **Infinite Loop**, came from to begin with? Many have trod down this path before you including me.

And that brings us to the final Action on the list, **Run Application**. I have already clicked upon the **Options** button to bring up the dialog form shown here in Figure 10-30. With this Action, your Event may ◉ **Run Applications** or ○ **Execute a DDE Command**. That is to say, launch an external application or execute a Dynamic Database Exchange (DDE) command.

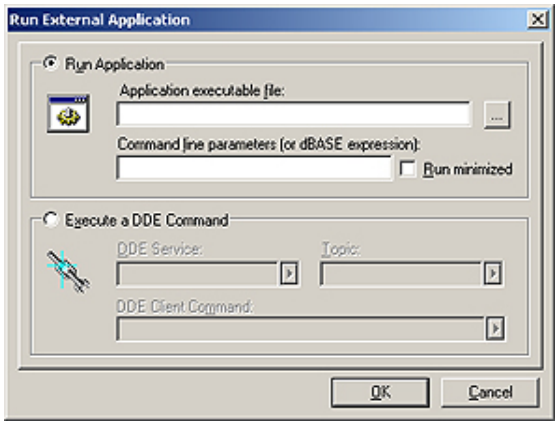


Figure 10-30

Tip

*Use your head. These are Automated Processes with the emphasis on them being **Automated**. Now don't go and ask your Event to run an external application that requires manual intervention. That is just a contradiction to automated.*

If you choose the default, ◉ **Run Application**, then in the **Application executable file:** field, you must point to the executable. I find it easier to browse to the executable by clicking on the ellipses

(**...**) button at the end of the field. This way I'm certain that the resulting path is correct, and formatted exactly as GoldMine would want it to be. It might look something like this:

Y:\GoldMine\ClipDate\ClipDate.exe

Be very careful. Know from where your Automated Processes will be scanned. For Instance: Just because the mapped drive Y:\ is visible from my development computer does not mean that it will be mapped or visible from the computer that is performing the Automated Process scans. You should make an effort to have any executables, that will be run via Automated Processes, installed under the networked GoldMine folder. This way, you will be certain that the executable is visible no matter which networked computer is scanning the Processes.

Some external applications will accept **Command line parameters (or dBASE expressions):**, and, if the particular external application will accept them, they are to be entered via this field. For Instance: Let's say that you had a link ID to another datatbase in the GoldMine Key5 field, and you wanted to pass that as a parameter:

[/u:] + trim(upper(Contact1->Key5))

That might just do the trick for you.

I'm really hesitant to talk about the ○ **Execute a DDE Command** option as I have never had the opportunity to test this out myself. I will give you the information as supplied from the GoldMine Help files, and maybe you can make more out of it than I could.

To send DDE commands:

1. Select **Execute a DDE Command**.
2. Select a **DDE Service**.
3. Select a **Topic**.
4. Select or type a **DDE Client Command**.

   Note: Each product application uses a unique format for DDE requests. Consult that product's documentation for details.

5. Click **OK**.

Now, based on my programming experience, I can tell you that a DDE Service might be GoldMine. While a Topic might be Data. As to what a DDE Client Command is, I am totally at a loss, and hoping that one of you may supply the correct answer. What I do know, from my programming experience, against the GoldMine API, is that you must establish a channel first. You may then use that channel to pull other information from GoldMine.

For Instance: Here is the Visual FoxPro code to do just that:

```
* Open DDE conversation
    ThisForm.nCh = ddeInitiate([GoldMine],[Data])

* Retrieve Data
    ThisForm.AccountNo = ddeRequest(ThisForm.nCh, [AccountNo])
    ThisForm.Company = trim(ddeRequest(ThisForm.nCh, [Company]))
    ThisForm.Contact = trim(ddeRequest(ThisForm.nCh, [Contact]))
    ThisForm.SysDir = ddeRequest(ThisForm.nCh, [&SysDir])
    ThisForm.GMSystem = upper(left(ThisForm.GMSystem, len(ThisForm.GMSystem)-1))
    ThisForm.GMBase = ddeRequest(ThisForm.nCh, [&GoldDir])
    ThisForm.Run_Date = ddeRequest(ThisForm.nCh, [Contact2->udRun])
    ThisForm.Run_Time = ddeRequest(ThisForm.nCh, [Contact2->ucRunTime])
    ThisForm.Contract_Time = val(ddeRequest(ThisForm.nCh, [Contact2->unCntTime]))

* Close DDE conversation
    ddeTerminate(ThisForm.nCh)
```

Now, as to how to translate that to the dialog form in Figure 10-30, I am at a complete loss. It is now time for you to step up to the bat. If you know how to correctly use this dialog form, then send me an e-mail to DJ Hunt <DJ@DJHunt.US>.

Now we have a new Action to discuss for this, the latest build of GoldMine Premium, and it is called **Create Case**. If you were to click on th **Options** button, you would bring up the dialog form shown on the next page in Figure 10-31.

You only have 3 possible options here:

- ○ **Use Existing Case Template:**

  Create A New Case Template

- ◉ **Create an empty case.**

As I have no Case Templates available, the default on this dialog form would to ◉ **Create an empty case**.

We have reached the end, and there aren't any more Actions to be covered for this build of GoldMine Premium. I think that I have been able to cover all of the available Triggers, and the the available Actions in enough detail to show you just how intricate a GoldMine Automated Processes could become.

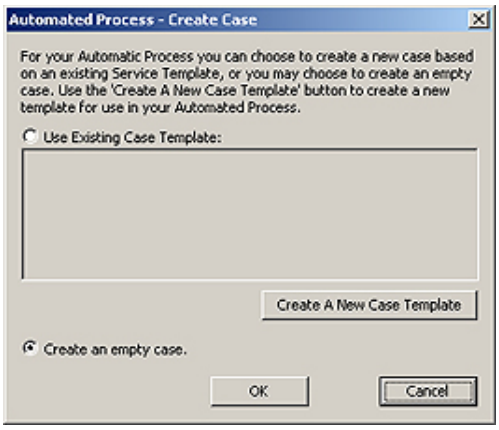


Figure 10-31

Tid Bits

I would like to continue on by looking at some of the subcomponents of events, and some related things that you could do that may help you achieve your goals through the use of Automated Processes.

Tid Bit 1:

As a rule, when creating events, I personally make heavy use of **dBase Conditions** both as a **Trigger**, and as a **Trigger Filter**. I have found, more often than not, that the dBase expressions allow me much more flexibility. In this light, I find the Age (+/-days): for Scheduled Activity to be very confusing to most GoldMine Administrators. I, therefore, always leave this set to 0, and make heavy use of a Flag to indicate to me whether a Process was or was not triggered. Witness the example that was shown to you earlier in this chapter. The **Contact2.Comments** field is seldom used, and affords you 65 characters for use as 65 individual Flags. Additionally, in GoldMine, you can hide from display, any field that you want to, with the use of the -2 expression as I discussed with you in another chapter in this book. Therefore, you could use your Comments field as your Flags area, and you could hide the Comments field, displayed on the Summary tab by default, from view. Alternatively, you could leave the field displayed, but make sure that only the Master user has Update Rights under the Security area for that field.

Tid Bit 2:

Tip

Where possible program the Lookup.ini to auto populate these field upon the creation of a new contact record.

Another item, that I always advocate, is the use of user defined fields to hold the UserID for Events that will be creating a scheduled or a historical activity. In either of these activities there are radio button selections for the User. One of those options is to take a ◉ **User from field**. I may even employ a number of user defined fields as different events may warrant the creation of activities for different users such as a Manager, an Account Representative, etcetera. I have one client with 12 user defined UserID fields for all of the Processes that they utilize. I usually place these fields on an Administration screen that is visible only to those users possessing Master Rights.

Tid Bit 3:

Did you know, when creating a scheduled activity, or a historical activity, that you can use an expression in the **Reference:** field that will be processed at the time the activity is created? Yes, I think that I mentioned this earlier. I've worked with a client creating the example that I will use here. Create an Event or modify an existing Event. Move to the Action page of the Wizard, and set **Perform Action:** to **Schedule Activity**. Click on the **Options** button, and then again on the **Activity Details** button. In the **Reference:** field you should enter:

```
[New Appointment on: ]+ Wdate(Contact2->MeetDateOn,3)
```

That's it. When this activity is scheduled through Automated Processes, the **Reference:** field will be populated, and displayed where appropriate. Here is an example of the results:

```
New Appointment on: Wednesday, October 3, 2007
```

This will add more detail to the created activities, viewable on the users Day Calendar, as created via an Automated Processes.

Tid Bit 4:

Okay, if you've understood nothing else in this chapter, pay particular attention to this. Whether you are a GoldMine Consultant, a GoldMine Administrator, or even a GoldMine Enduser, you should adhere to these steps when developing your Automated Processes.

Step 1: Type up the process steps in Microsoft Word.

Step 2: Relate those process steps to GoldMine Event Actions.

Step 3: Determine what Trigger/Trigger Filter is required to cause the Action to be applied for each step, and insert it into Word preceeding the Action.

Step 4: Leave a space or two between the Action, and the following Event.

Step 5: Determine if there are any Preemptive Events that are required for things such as possibly, removing the process before it has completed all of its Sequential Events.

Step 6: Create your Process in the Automated Process Manager.

Step 7: Test your Process thoroughly against a single record.

Step 8: Create a Preemptive Event in your Observer Process that will cause this Event to be attached to the record at the appropriate time.

Step 9: Test the Observer Process/Your Process functionality against a single record again.

Step 10: Match your Word document to the final Process that you have created, fixing anything in the document that may have changed while actually building the process. Your goal is a detailed document that describes all of the Triggers, and Actions completely that you can refer to a year from now when you have to make changes to this Process.

Step 11: Copy the text of your completed Word document into your GoldMine KnowledgeBase for easy reference in the future.

Tid Bit 5:

In Chapter 3, I discussed the ini setting to display the Automated Process flow in the GoldMine Process Monitor. It is so important in the Automated Processes development phase that I will reprint it here:

■ **Programmers - Display AP Flow within the GoldMine Process Monitor**

There has been a need to debug an Automated Process for a long time now. To facilitate this, FrontRange had added the ability to switch On/Off the flow of the processes through the GoldMine Process Monitor. This is done individually within the UserID.ini or as a Corporate Override in the GM.ini (not recommended).

Note

Should you set this option to display Automated Process flow, you want to make certain that you remove the option when you are done with your AP testing. This has been known to cause processing issues in the past.

This is the switch statement:

```
[GoldMine]
APDebugLog=1
```

This is a typical result of said action:

```
0[1] Automated Processes  [1:12 pm - 7/24/2007]
0[1] Scanning Contact: Computerese; DJ Hunt
0[1] Read Process:  Update Department Field  [Next Event: 100, Process: 9G4RA45$W=98R#Y]
0[1]    100   Test Update Field
0[1]            --> Triggered.
0[1]                Update Field: DEPARTMENT with DJ Hunt
0[1]     Remove ended process: 9G4RA45$W=98R#Y
4[1] Automated Processes: 1 Contacts; 1 Scanned; 1 Triggered; 1 Pass(es)   [Ended 1:12 pm -
       7/24/2007; Dur: 0:00]
```

As you can see this could be valuable information when trying to debug an Automated Process. Of course, if you are perfect, and never make a mistake when developing your AP's this is, naturally, not required.

Wrap Up

This concludes my discussion of Automated Processes. There are a couple of things that I would like to reiterate. Again, I would like to state that you should take baby steps with Automated Processes. Solve one Event at a time, and then put it into action using the Observer Process as the launching point for all other Processes.

Lastly, and most importantly, you must have a process in order to automate a process. That only sounds logical doesn't it? GoldMine does provide you with some examples under the **(public)** User Group. All of these examples must be attached to the appropriate record by hand. They do not make

use of the Observer Process paradigm as I have discussed it in this chapter. They are, however, good examples of Processes, and expose you to a typical Process flow. These GoldMine Processes expose a wider range of the use of the available options than I may have exposed here.

It was my aim to expose you to a technique that I have employed over the years, as have many readers of my prior book "**Sales Force Automation with GoldMine 4.0**", which has been employed with success over the years. Sales Force Automation with GoldMine 4.0 was called the blueprint for GoldMine Automated Processes. The techniques discussed are still applicable to the GoldMine Premium product. You may still purchase Sales Force Automation with GoldMine 4.0, as an eBook only, and only from DJ Hunt <DJ@DJHunt.US>.

11

In This Chapter

GM+View, was new way back in GoldMine 6 BCM, and allows users to view HTML documents directly within GoldMine. GM+Browser was new to GoldMine 6.7, and basically does the same as GM+View, however, in a detached Window, if in Windowed mode, or as an independent Tab, if in the Tabbed mode. There are many, many different presentations that one could define for a given GoldMine installation, and the choice is always yours. You might want to display a location map, a contact resume, the weather at the contacts location, or any of a myriad of other possibilities. One of the more popular GM+Views/GM+Browser is to display linked data from an external table, and I will show you how to accomplish that using GoldMines own **ContHist** table as my external table. However, you can apply this same technique to any external table for which you have an ODBC or OLE connection capability. Just wait, you'll see.

I have supplied you with many different templates that you can just attach to your GM+View/GM+Browser that should work just fine with little or no modification. The zip file containing these is embedded in the zip file that you downloaded when you downloaded the eBook version of this book. If you purchased this book through a print on demand service, you must e-mail me a copy of the **Shipping Order** included within your book shipment. You will then receive an e-mail back from me allowing you to download the eBook version with all of these templates. Buy one, get two. That e-mail address would be: **DJ@DJHunt.US**. Please make sure that the **Subject:** of the e-mail message is **eBook Request**, and I will get them right out to you.

I'll discuss the implementation for most of the templates later in this chapter, however, there are many where the implementation is identical to a previously discussed template, and I will not be discussing these as the process for attaching and utilizing the template will be identical. It will be up to you to make the rest of the templates works.

One thing that I want to discuss with you right off the bat is the **GM+Browser** which, in my opinion, is a great bonus. The GM+Browser is a free-floating browser that defaults the display to the contacts website, however, you can easily add any of your GM+Views to this browser view. Figure 11-1 shows

Figure 11-1

how one could display specific history items for the active GoldMine Contact record in its own screen. Take a look. You'll see this same **Contract History**

GM+View later in this chapter. As a bonus, however, if using the windowed mode for GoldMine, you can leave the GM+Browser window open in your GoldMine frame displaying the current history activity with notes if you would like. Very nice indeed, although you may have to work around the current refresh issues of the GM+Browser.

First, let's discuss some of the configuration features that are behind GM+Views. From the GoldMine menu, click on **Web** | **Setup GM+View...** to bring up the dialog form shown here in Figure 11-2.

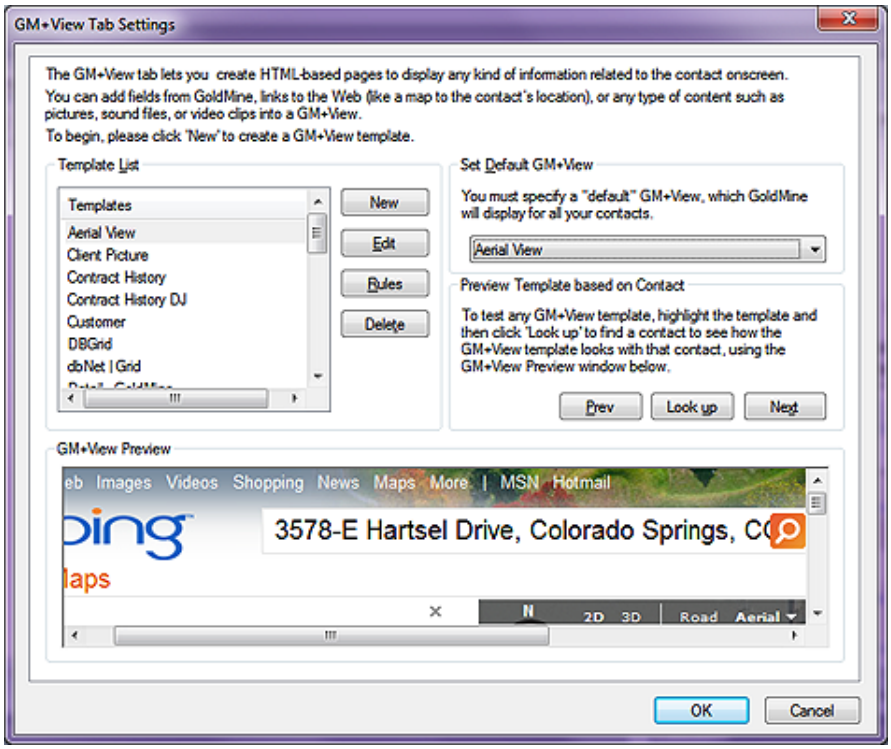


Figure 11-2

Naturally, yours may not look exactly like this one if your GoldMine is not a new installation. For instance, your new GoldMine installation might not have the **Car**, and the **Property** templates displayed, whereas this is a screenshot from my own GoldMine. If you do, however, have these templates, I would immediately ask you to **Delete** the **Car**, and the **Property** templates so that we may begin our discussion of GM+Views with you having a clean slate on your GoldMine. On the other hand, maybe you would want to keep those for a while as they are also good examples for you to fall back on for information.

Rules

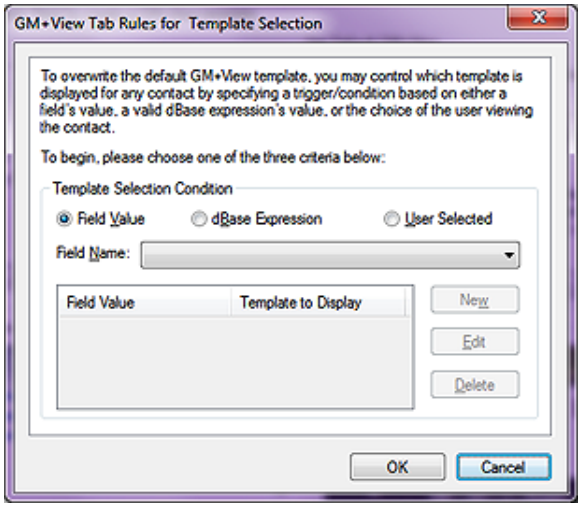


Figure 11-3a

Let's first begin by examining the **Rules** that one may set up. Click on the Rules button to bring forth the dialog forms shown in Figures 11-3a, 11-3b and 11-3c.

You will notice, here in Figure 11-3a, that the first radio button, ◉ **Field Value**, allows you to default display one, and only one, GM+View per Contact record based upon a value contained in any Contact1 or Contact2 table field. The different GM+Views must be pre developed before you can set this rule into action. The activity of creating a rule here is pretty self-explanatory, however, I will take a little time to go over the basics.

You must first select a **Field Name:** from the available fields on the drop list, which as we already know, is all of the fields contained in the Contact1/Contact2 tables. This is the field that must contain the value that will instruct GoldMine as to which GM+View to display when a certain value is contained within this field. You may have read about this earlier in this book. It was called Record Typing.

You would then click on the **New** button, and associate a **Field Value** with a GM+View (**Template to Display**). That's all there is to it. Naturally, you could **Edit** the entry, or **Delete** the entry if the opportunity arises.

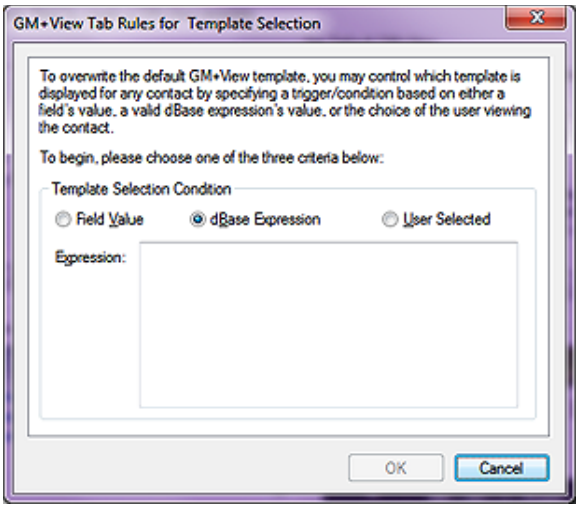


Figure 11-3b

In Figure 11-3b, you can see that an **Expression:** field is exposed when one selects the O **dBase Expression** radio button. It is possible to use nested **iif()** functions (refer to Appendix A), as I described for you in Chapter 4 when we were coloring fields or changing label colors. Again, this type of rule is limiting. You can only display one GM+View per Contact record at a time. This goes back to my discussion on Record Typing that I presented to you in Chapter 4.

An example of an expression that might be utilized in this area would be something on the order of:

```
iif(upper(trim(Contact1.Key1))==[GM DEALER], [Yahoo Map], [Yahoo Weather])
```

I feel that there is too much value under the **GM+View** tab to limit it to just one view per contact record. I advocate creating a number of views, and allowing them to be O **User Selected** by selecting this radio button, as shown in Figure 11-3c. That's all that is required under **GM+View Tab Rules for Template Selection**. Now the user will be able to right-click under the GM+View tab, and select from any of the developed html documents.

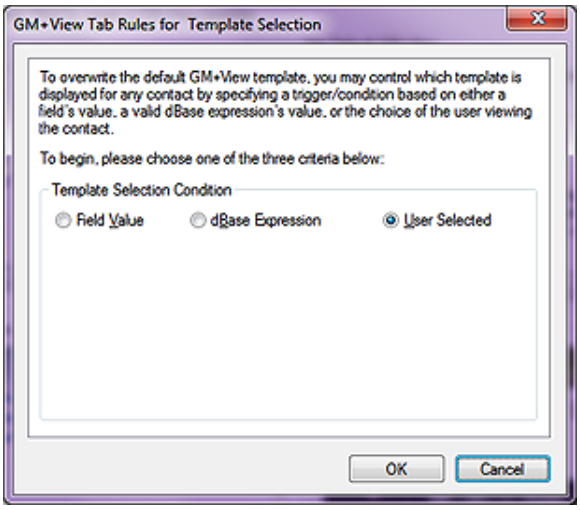


Figure 11-3c

Templates

I have pre developed a handful of GM+Views for you to use, and zipped them into **GM+View HTML.zip** which are included in the eBook download. If you have purchased a Print on Demand or eBook copy of this book, then you may send an e-mail to: **DJ@DJHunt.US** with a copy of your receipt. Please make sure that your **Subject:** is **eBook Request**. I will then send you the download information for the eBook, and all of the templates discussed in this book.

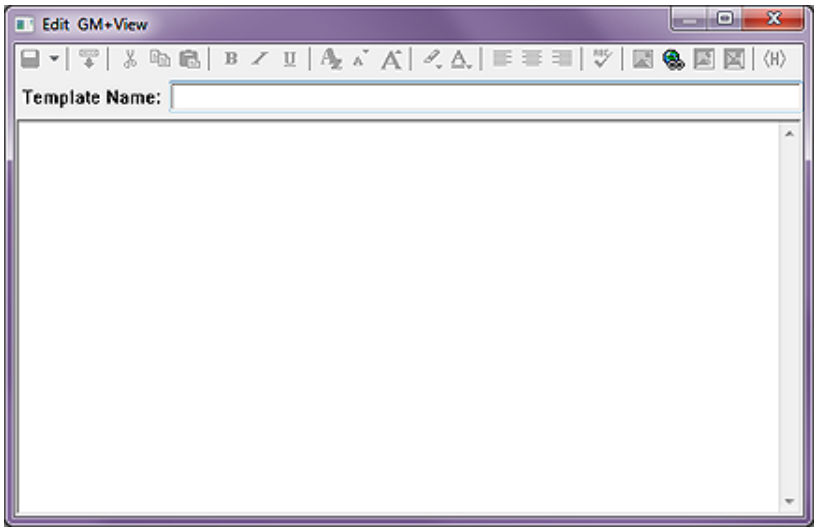


Figure 11-4

Note

*There are several templates included in **GM+View HTML.zip** that are not discussed in this chapter, however, I highly recommend that you examine them, as they may be useful for your organization. They are all added to the **GM+View** in the same manner as will be discussed in this chapter. Most require no modification.*

Note

*Whenever I talk about editing the HTML code of a view, I mean to do so in anything other than the GoldMine HTML Editor. Even the lowly **NotePad** application is better for editing HTML than is the GoldMine HTML Editor.*

I am now going to discuss creating GM+View templates using these examples. For my first creation, I will be displaying a Word document file that exists on the world wide web. For this example I'll display a resume.

First I would ask you to click on the **New** button, as was previously shown to you in Figure 11-2, to bring up the standard GoldMine GM+View Editor as shown on the previous page in Figure 11-4.

Later in this chapter, I will give you one example of creating a GM+View template, within the GoldMine Editor, to display linked pictures. It is my practice to create all of my views in Adobe Dreamweaver, and then to include them into my GoldMine GM+View template. I will discuss that process now, which will be the same for all of the document templates that I have supplied to you. When necessary, I will discuss changes that need to be made to the HTML code in my supplied views for them to function best for you.

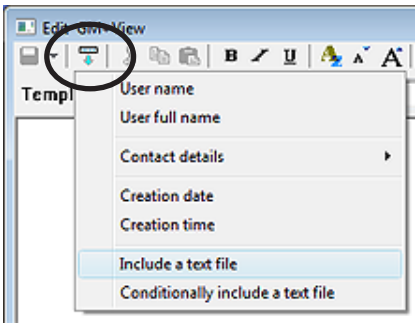


Figure 11-5

Here in Figure 11-5, you will notice that I have highlighted an icon. This icon will become available to you after you have moved your cursor to the body area. Memorize this icon as we will be using it often, as well as the **Include a text file** option that clicking upon it exposes. As I explained in the Tip, you can also use the same technique when creating your E-Mail Templates. This option, under this icon, should be your most utilized tool.

■ Displaying the same document for all contacts

In Figure 11-4 on the previous page, you can see a field called **Template Name:** which is where one would enter, you guessed it, the name of the template. For this one, I ask you to enter **Display Document**, and then tab to the body area. This is the name that will appear in the GM+View list. I will give you a warning, you are limited to 19 characters in this field, and you want to make sure that each GM+View has a unique name.

Notice that, once in the body, certain icons become enabled. I am only interested in one, the second icon from the left, Figure 11-5 (encircled), which, when you hover over it, has a pop up ToolTip that states: **Insert Field**. If you click on this icon, you will be presented with a local menu from which you will want to select: **Include a text file**. This action immediately instantiates an **Open** dialog form from which you would be asked to find: **Display DOC.html**, which I hope that you unzipped to an HTML folder under your GoldMine folder.

In my setup this action inserted:

```
<<file:Z:\Data\Book The Definitive Guide to GoldMine Premium\HTML Docs\Display DOC.html>>
```

It can't get any simpler than that. Now close this template, and save it. That's all, however, I would warn you to read the sidebar which applies to displaying documents or pdfs. When you save this type of template, you may be asked to **Open** the document. Don't. Just **Cancel** that.

■ Displaying a different document for each contact

Should you want to have a different document per contact, then I would pull the relevant string out of my HTML document, and stuff it into a GoldMine **Contact2** field. I used **UserDef01**, and expanded the **UserDef01** field length to 250 characters for this example.

Here is the relevant statement taken from the template:

```
http://www.DJHunt.US/DWSite/eBook/DJ%20Hunt%20Resume.doc
```

I have pasted this string into the **UserDef01** field that I'm displaying on my screen. It is important that you have the string exactly as displayed only inserting your path, and document name with extension. Notice that I am storing my document, in this case, on my website.

I now create a **New** template in GM+Views with a new **Template Name:**, maybe **Display Doc/Contact**. This time, when I enter the body, I simply type click on the **<H>** icon and I paste this code from my HTML Editor:

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">

        <head>
```

Tip

*This same logic applies to **E-mail Templates** as it does to the **GM+View Templates**. I never use the **GoldMine HTML Editor <H>** for creating my templates. I always create my templates with a real HTML Editor, and simply **Include a text file** in the body of the GoldMine E-mail Template. **Do Not** even consider a **Copy/Paste** from your editor to the GoldMine editor as an option unless you wish to do this on every edit of this template.*

Note

Don't be afraid to look at the coding of the HTML documents that I have supplied you. Use your favorite HTML Editor, Microsoft Expressions, Adobe Dreamweaver, or even Windows Notepad to view the underlying code. You'll be able to create your own documents in the same manner, and then point to them from within GoldMine.

WARNING

If you are using Office 2007, I have noticed that the Internet Plug-in for Word is not being installed, hence, my document, instead of appearing under the GM+View tab, is opening up directly into Microsoft Word 2007. Read that as very slow.

Note

Please note that this could have been a local path as opposed to a URL, and it would have worked just as nicely.

Further, you could have had the Lookup.ini populate this using an expression, and trigger upon NewRecord creation if you wish.

Note

*Did you notice how nicely structured my document is thanks to my HTML Editor? Now compare that to that which you just say when you clicked on the **<H>** icon to get into the GoldMine HTML Editor.*

Oh, wait! There is no real comparison now, is there?

Note

Specifically notice the highlighted text which defines the GoldMine field which contains the variable data per Contact record.

Note

*This structured code was taken from-**Display DOC_2.html** as distributed in the template collection for this book.*

WARNING

If your system is configured to display the file in the Internet Explorer window as opposed to launching the document application, there is a little flaw in the system, one must click off of the GM+View tab when changing records, and then back on to refresh the record. It does work, and that is what counts.

Note

Statement is wrapped for presentation only, and would be one continuous statement within the Lookup.ini.

```
<!--
  This GM+View Template is provided as part of GoldMine Premium - The Definitive Guide
  series of books, and the author is providing this without guarantee or additional support.
-->

        <meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1" />
        <title>GM+View Document</title>

        <SCRIPT language=JavaScript>

                <!--
                function redirect()
                {
                 window.location = "&lt;&lt;trim(Contact2.UserDef01)&gt;&gt;"
                }

                setTimeout("redirect();", 0000)
                // -->

        </SCRIPT>

    </head>

    <body>
    </body>
</html>
```

I then save this template. Give it a try. Again this type of view works best if there is but one view per contact record.

I would like to add that this would be an excellent use for the Lookup.ini. You could update the UserDef01 field string with the Contact name or Company name, as you choose, to repoint to a different document for each contact record. Here is an example of how that specific Lookup.ini code might look:

```
[AutoUpdate]
  NewRecord = UserDef01

[UserDef01]
  Otherwise = &" http://www.DJHunt.US/DWSite/eBook/" + &FirstName + "%20" + &LastName
              + "%20Resume.doc"
```

This could prove to be extremely useful as you could have an entire book per contact record by following the steps as described in ■ **Displaying a different document for each contact.** You might ask, "*Why would this be useful?*". If you had ever used **Notes** to store historical information for a contact, and found that you were losing notes when you exceeded the limit, you wouldn't be asking that question. This is a space to store notes that would be virtually unlimited. When combining this with the Lookup.ini, well, you can just imagine all of the possibilities.

■ Displaying the same pdf for all contacts

The next template that I would like you to try is the **Display PDF.html**. For this template, I simply follow the same steps as discussed previously for ■ **Displaying the same document for all contacts**. This will only work properly, within GM+Views/GM+Browser, if the users have the Adobe PDF plug-in for Internet Explorer 8 installed on their system. I gave mine a **Template Name:** of **Display PDF**. Where as, I used to state: "*Refer to the sidebar warning as it is applicable for a PDF view as well as the DOC views when the plug-in is installed.*", in GoldMine Premium, I no longer need to make this statement. Although you may still right-click to produce your list of available GM+Views, and this is the component that used to conflict with the PDF plug-in, the developers have now added another navigation utility, the drop list in the GM+Views header bar as highlighted in Figure 11-6.

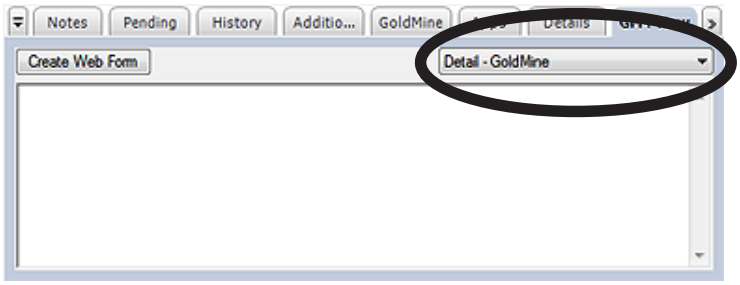


Figure 11-6

Oh, the headaches that this saves.

■ Displaying a different pdf for each contact

Well, if it holds true for Documents then it should hold true for PDFs as well, and it does. Should you want to have a different document per contact, then I would pull the relevant string out of my HTML document, and stuff it into a GoldMine Contact2 field. I used the UserDef01 field again, as its length has already been expanded to 250 characters, for this example.

Here is the relevant statement taken from the template:

```
http://www.DJHunt.US/DWSite/eBook/DJ%20Hunt%20Resume.pdf
```

I now create a **New** template in GM+Views with a new **Template Name:**, maybe **Display PDF/Contact**. This time, when I enter the body, I simply type click on the **<H>** icon and I paste this code from my HTML Editor:

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">

        <head>

        <!--
          This GM+View Template is provided as part of GoldMine Premium - The Definitive Guide
          series of books, and the author is providing this without guarentee or additional support.
        -->

                <meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1" />
                <title>GM+View Document</title>

                <SCRIPT language=JavaScript>

                        <!--
                        function redirect()
                        {
                         window.location = "&lt;&lt;trim(Contact2.UserDef01)&gt;&gt;"
                        }

                        setTimeout("redirect();", 0000)
                        // -->

                </SCRIPT>

        </head>

        <body>
        </body>
</html>
```

I then save this template as before, and give it a try. Refer to all of the previous WARNINGs and Notes governing DOCs. Again this type of view works best if there is but one view per contact record.

I would like to reiterate that this would be an excellent use for the Lookup.ini. You could update the UserDef01 field string with the Contact name or Company name, as you choose, to repoint to a different document for each contact record. See my previous DOC example.

■ Displaying the Primary Website for your contact record

The next template that I would like you to try is the **Display Web.html**. For this template, I will again ask you to follow the same steps as discussed on the previous page for **■ Displaying the same document for all contacts**. There now, wasn't this simple?

■ Displaying Google Driving Directions to the contact location from a fixed location

The next template that we are going to look at is the **Google Directions.html**. For this template I have decided to utilize the registered license information that GoldMine maintains as my starting point. Later, under MapQuest directions, you will see that I have hard coded my starting point. We, again, follow the same steps as discussed previously for **■ Displaying the same document for all contacts**. Here is the driving code section from this template:

```
window.location = "http://maps.google.com/maps?q=&lt;&lt;&amp;licinfo_address1&gt;&gt;%20&lt;&lt;&amp;licinfo_zip&gt;&gt;%20to%20&lt;&lt;&amp;address1&gt;&gt;%20&lt;&lt;&amp;zip&gt;&gt;"
```

You will notice that I have highlighted the code in this statement where the GoldMine registered information is being taken. With Google, we only need to provide minimal information to acquire

Note

***HTML** = Hypertext Markup Language*

Note

I can't emphasize enough the need to test your GM+Views before deploying they. I will have spent well over 10 hours testing all of the GM+Views templates that I have created for this book, however, that does not relieve you of your responsibility to test them within your environment.

WebSites, and their parameters are constantly in flux. You should validate your GM+Views regularly.

maximum results. Very slick, and maybe the reason many prefer Google over the MapQuest solution that I will provide to you shortly.

■ Displaying a Google Location Map

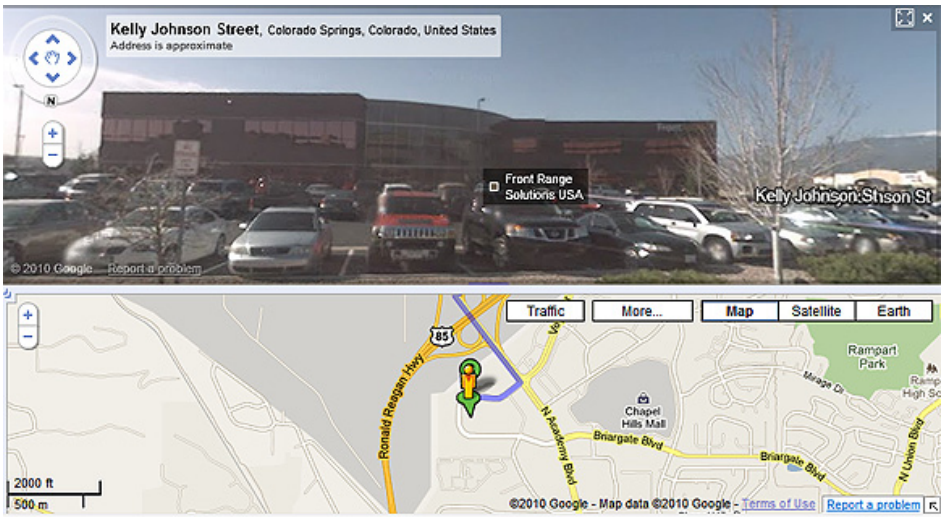


Figure 11-7

In Figure 11-7, I thought that I would show you a screenshot of what one of these templates could produce for you in either GM+View or GM+Browser. Each organization has their own unique needs, and these templates may not be of any use to you at all. I have, however, tested them all, and, at the time of my testing they were working nicely. For **Google Location.html** the control code is:

```
window.location = "http://maps.google.com/maps?oi=map&amp;q=&lt;&lt;&amp;Address1&gt;&gt;%20
&lt;&lt;&amp;City&gt;&gt;%20&lt;&lt;&amp;State&gt;&gt;%20&lt;&lt;&amp;Zip&gt;&gt;"
```

You can see, highlighted, that we are pushing the GoldMine Contact record **Contact1.Address1**, **Contact1.City**, **Contact1.State**, and **Contact1.Zip** to Google for its resolution into a location map.

■ Displaying a Google Search

I know that you have used this Google feature thousands of times in the past. The Search Engine was where Google got its beginnings. From this code:

```
window.location = "http://www.google.com/search?as_q=&num=100&hl=en&ie=UTF-8&oe=UTF-
8&btnG=Google+Search&as_epq=&lt;&lt;&Company&gt;&gt;&as_oq=&as_eq=&lr=&as_ft=i&as_
filetype=&as_qdr=all&as_occt=any&as_dt=i&as_sitesearch=&safe=images"
```

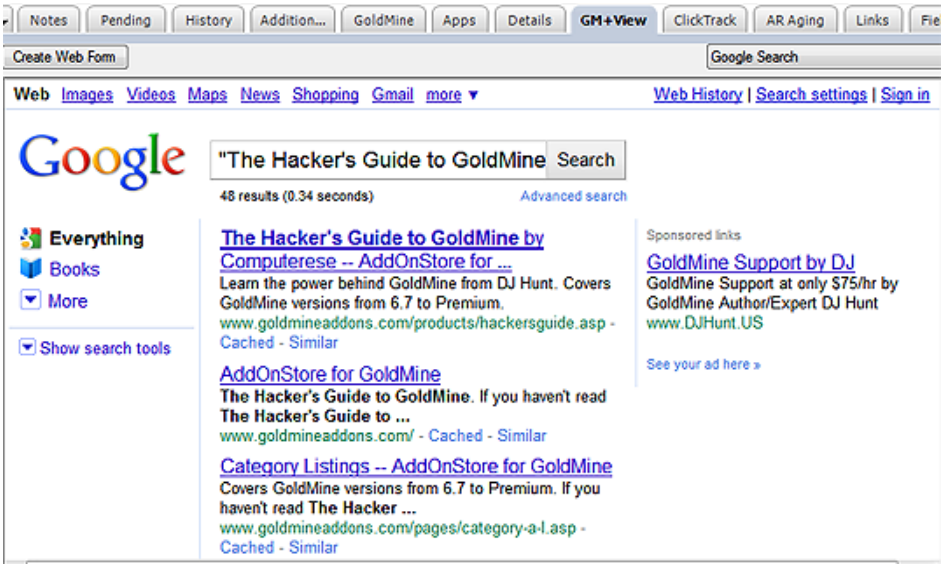


Figure 11-8

You supply only the Contact record Company information to the Google Search Engine, and your GM+View becomes your new active Search Engine, Figure 11-8.

■ Displaying a Google Group Search

This is much the same as a Google Search, only this time we are searching with slightly different criterion, and using a slightly different part of the Google Search Engine. The primary code is:

```
window.location = "http://groups.google.com/groups/search?lr=&safe=off&num=10&q=&lt;&lt;&amp;
Company&gt;&gt;+&lt;&lt;&amp;City&gt;&gt;&safe=off&qt_s=Search"
```

■ Displaying MapQuest Driving Directions - Fixed Starting Location

The next template that I would like you to try is the **MapQuest Directions.html**. For this template, we again follow the same steps as discussed previously for ■ **Displaying the same document for all contacts**, however, you will need to modify the MapQuest Directions.html document slightly. That is unless you want to know how to get from my location to your clients location. Open MapQuest Directions.html in your favorite HTML editor (**not** Microsoft Word or Publisher). The line that you would be looking for is:

```
window.location = "http://www.mapquest.com/directions/main.adp?go=1&1a=150+PRATT+RD
&1c=FITCHBURG&1s=MA&1z=01420-4142&2a=&lt;&lt;&amp;&Address1&gt;&gt;&amp;&2c=&lt;&lt;
&amp;&City&gt;&gt;&amp;&2s=&lt;&lt;&amp;&State&gt;&gt;&amp;&2z=&lt;&lt;&amp;&Zip&gt;&gt;"
```

It is important to note that the above is a single line of text. I have highlighted the items that you will need to modify before saving the document, however, I would like to point out some things. The plus sign (**+**) is the equivalent of a space between words for MapQuest. There should not be spaces, carriage returns or line feeds in this statement anywhere. Thus, my information, in the above statement, which is:

```
150 PRATT ROAD
FITCHBURG
MA
01420-4142
```

...is represented as:

```
150+PRATT+ROAD
FITCHBURG
MA
01420-4142
```

Of course, you would replace my addressing information with your own MapQuest Starting location. Being in the United States, I have not had the opportunity to test this in other countries, however, I suspect that you would certainly need to add the country code to this script if you were in a different country. There are other services available for other countries, and you may be able to use this code as the basis for setting up your own script. After you've made your changes, save them, and then test this document in your GM+Views.

Note

*To **Copy**, and **Paste** location maps in GoldMine E-mail, you must be have to Option selected to **Use HTML when creating E-mail**.*

■ Displaying a MapQuest Location Map

The HTML document that you will use for this one is **MapQuest Location.html**. Again, no changes should be required for this document to produce the desired results. Just follow the steps that I discussed previously under ■ **Displaying the same document for all contacts**. I gave this template a **Template Name:** of **MapQuest Local Map**. This is rather neat in a way. Under the GM+View tab, with this view selected, I click on the Printer-Friendly hotlink. Then I right-click on the results, choosing **Select All**, and then right-clicking again, choosing **Copy** from the local menu. I can then **Paste** this map into an E-mail to be sent to a representative or anyone requiring this information. Additionally, as of today anyway, I could display this as a Street map, Aerial map, I could show Gas Stations, Parking Garages, and more.

■ Displaying Yahoo Driving Directions

So as not to show favoritism, I have included some Yahoo GM+Views as well. The first of which is the HTML document that you will use for this exercise is **Yahoo Directions.html**. No changes should be required for this document to produce the desired results if your starting point is to be the same point as indicated in your GoldMine Registration information. You may want to use hard coded starting locations, and this should be an easy modification for you at this point. Make sure that you utilize a good HTML Editor, and that you **do not** use Microsoft Word or Microsoft Publisher. Just follow the steps that I discussed previously under ■ **Displaying the same document for all contacts**. I gave this template a **Template Name:** of **Yahoo Directions** within my GoldMine.

The key script line in this document is:

```
window.location = "http://maps.yahoo.com/py/ddResults.py?Pyt=Tmap^&tarname=^&tardesc=^&newname=^&newdesc=^&newHash=^&newTHash=^&tlt=^&tln=^&slt=^&sln=^&newFL=Use+Address+Below^&newaddr=&lt;&lt;&LicInfo_Address1&gt;&gt;+^&newcsz=&lt;&lt;&LicInfo_City&gt;&gt;%2C+&lt;&lt;&LicInfo_State&gt;&gt;+&lt;&lt;&LicInfo_Zip&gt;&gt;+&newcountry=us^&newTFL=Use+Address+Below^&newtaddr=&lt;&lt;contact1-&gt;address1&gt;&gt;+^&tcsz=&lt;&lt;contact1-&gt;city&gt;&gt;%2C+-&lt;&lt;contact1-&gt;state&gt;&gt;+&lt;&lt;contact1-&gt;zip&gt;&gt;+&tcountry=us^&Submit=Get+Directions"
```

The words that are highlighted (all begin with &LicInfo for you printed edition readers) are the portions of the code that would need to be changed if you were to hard code the starting point. See the example that I previously discussed back under the ■ **Displaying MapQuest Driving Directions - Fixed Starting Location** section of this chapter.

■ Displaying Yahoo Location Map

The second Yahoo GM+View that I would like to present, and is the HTML document that you will use for this exercise is **Yahoo Location.html**. No changes should be required for this document to produce the desired results. I gave this template a **Template Name:** of **Yahoo Local Map** within my GoldMine.

The key script line in this document is:

```
window.location = "http://us.rd.yahoo.com/maps/us/insert/Tmap/extmap/*-http://maps.yahoo.com/maps_result?addr=&lt;&lt;&amp;address&gt;&gt;&amp;csz=&lt;&lt;&amp;CityStateZip&gt;&gt;&amp;country=us"
```

Those of you who are not located in the US, may want to change the country designation.

■ Displaying Yahoo Yellow Pages

Note

With the merger of Bing and Yahoo Searches, the Yahoo Yellow Pages script is pulling in aerial shots of this location. You may want to try this on your own to see what it will pull when you are ready to utilize it.

Remember, these sites are in constant flux, and you will have to modify your GM+Views occasionally.

The next Yahoo GM+View that I would like to present, and is the HTML document that you will use for this exercise is **Yahoo Yellow Pages.html**. No changes should be required for this document to produce the desired results. I gave this template a **Template Name:** of **Yahoo Yellow Pages** in my GoldMine.

The key script line in this document is:

```
window.location = "http://yp.yahoo.com/py/ypResults.py?stx=Savings+Bank&stp=a&tab=B2C&addr=&lt;&lt;&amp;Address1&gt;&gt;=&lt;&lt;&amp;City&gt;&gt;&state=&lt;&lt;&amp;State&gt;&gt;&zip=&lt;&lt;&amp;Zip&gt;&gt;&uzip=&lt;&lt;&amp;Zip&gt;&gt;&country=&lt;&lt;&amp;Country&gt;&gt;"
```

This script was pushing that we want to search for Savings Banks that are local to the Contacts address. As Country is a parameter, I suspect that this script should work nicely outside of the US as well as in the US.

■ Displaying Location Aerial Photo

The HTML document that you will use for this exercise is **Aerial Map.html**. No changes should be required for this document to produce the desired results. Just follow the steps that I discussed previously under ■ **Displaying the same document for all contacts**. I gave this template a **Template Name:** of **Aerial Photo** within my GoldMine.

Note

I would mention that the Hoover's template does require that you have a subscription to Hoover's as this is a paid service.

■ Displaying Hoovers Information

The HTML document that you will use for this exercise is **Hoovers.html**. No changes should be required for this document to produce the desired results. Just follow the steps that I discussed previously under ■ **Displaying the same document for all contacts**. I gave this template a **Template Name:** of **Display Hoovers** within my GoldMine.

■ Displaying the Weather for a Location

The HTML document that you will use for this one is **Local Weather.html**. Again, no changes should be required for this document to produce the desired results. Just follow the steps we discussed previously under ■ **Displaying the same document for all contacts**. I gave this template a **Template Name:** of **Display Local Weather**.

■ Displaying MultiMap Information (specifically designed for my UK readers)

The HTML document that you will use for this exercise is **UK MultiMap.html**. No changes should be required for this document to produce the desired results. Just follow the steps that I discussed previ-

Note

As of todays testing, MultiMap appears to have been gobbled up by Bing, and is now utilizing the Microsoft Maps.

ously under ■ **Displaying the same document for all contacts**. I gave this template a **Template Name:** of **UK MultiMap** within my GoldMine.

The relevant script is:

```
window.location="http://www.multimap.co.uk/map/browse.cgi?client=public&db=pc&addr1=&client=public&addr2=&advanced=&addr3=&pc=&lt;&lt;&amp;Zip&gt;&gt;&quicksearch=&lt;&lt;&amp;Zip&gt;&gt;";
```

You could easily modify this script to be more specific as I am only passing it the Zip Code (Don't you call this Postal Code?) information.

■ Displaying Your Social Networking Site Login

The HTML document that you will use for this one is **LinkedIn.html**. Again, no changes should be required for this document to produce the desired results. Just follow the steps we discussed previously under ■ **Displaying the same document for all contacts**. I gave this template a **Template Name:** of **LinkedIn**.

■ LinkedIn Search

Once you have logged in utilizing the ■ **Displaying Your Social Networking Site Login**, you may then utilize this LinkedIn Search for the Active Contact until you have logged back out. The HTML document that you will use for this one is **LinkedInSearch.html**. Again, no changes should be required for this document to produce the desired results. Just follow the steps we discussed previously under ■ **Displaying the same document for all contacts**. I gave this template a **Template Name:** of **LinkedIn Search**.

■ Displaying Active Windows Explorer

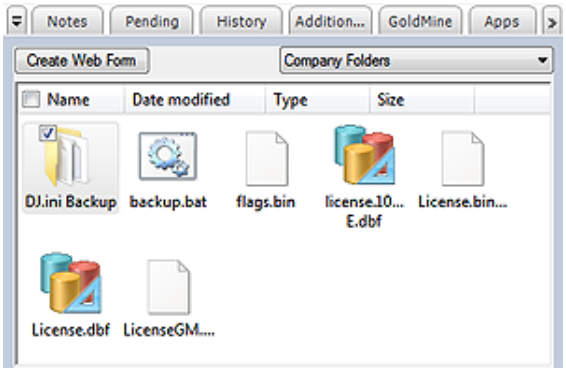


Figure 11-9

Scenario: You maintain a folder for each contact. In this folder are subfolders as well as documents and other applicable information about your contact. Each main folder is using the same name as a contacts name. You want to be able to view the different folders, and information contained in those folders from within your GM+Views. You want to actively open those documents from within your GM+Views.

The HTML document that you will use for this exercise is **Contact Explorer.html**. I gave this template a **Template Name:** of **Contact Explorer** in my GoldMine.

The relevant script is:

```
<meta http-equiv=REFRESH content="0;url=File:///\\Dell-Server\Apps\GoldMine\Temp\&lt;&lt;&amp;Contact&gt;&gt;\">
```

You may have noticed that the UNC path is: **\\Dell-Server\Apps\GoldMine\Temp**. It is important to understand that this must be a shared, and fully accessible path for everyone. You may then have noticed that I am pulling the contact name from the active record, therefore, against my record, the full UNC path when developed would be: **\\Dell-Server\Apps\GoldMine\Temp\DJ Hunt**. It is important to understand that the contact name may not contain any special characters that are not acceptable Windows pathing characters.

Figure 11-10

Passed Information

Number to Dial: **(978)342-3333**
Calling E-mail: **DJH@DJ-Hunt.com**
Caller Alias:
Send E-mail:
Number to Conference: **(978)342-3333**
Caller E-mail: **DJH@DJ-Hunt.com**
Caller Alias: **GoldMine Guru**

OrgID: **12345**
PinID: **67890**

Note

*These are older screenshots that I am reusing and my E-mail Address has since changed to: **DJ@DJHunt.US**.*

Pushing Information

This concludes the setup of the our predefined HTML documents. That wasn't too bad now, was it? This supplies you with a good taste for the potential of the GM+View/GM+Browser. I have also used GM+Views to **Pull** GoldMine data, and then, with the use of the **HTML Post** capability, **Pushed** that data to an ASP file to be processed at another location. In Figure 11-10 on the previous page, I show you a screen shot of this in action. This is one of several of the **HTML Post** uses that I have developed for clients.

Note

*Please note that I have not had the opportunity to test the **Submit** on this within GoldMine Premium 9.0.1.27 Beta build in a Windows 7 Ultimate environment, however, I have no reason to believe that it would still function as it had in the past.*

One of my clients' needed to push information from GoldMine to an Internet dialer, so I had to come up with a form that would do just that. The form shown previously in Figure 11-10 was the result of that exercise. The form itself is displayed here in Figure 11-11. Now I do want to emphasize that this is an **HTML Form**, and the **Dial Number** button has a form **Submit** action of:

"http://www.DJHunt.US/GMPush.asp"

Basically, all I am doing is taking the information from GoldMine, and placing it into this form. There are a couple of editable fields, however, the gist of the exercise is just to push the contact information out to an ASP file for processing.

You can see in Figure 11-11 just how I elected to capture the information, and in Figure 11-10, exactly what is being passed to the ASP file for the example given. There is nothing exotic about this exercise, however, it does show nicely how you can simply pass information from GoldMine to any website.

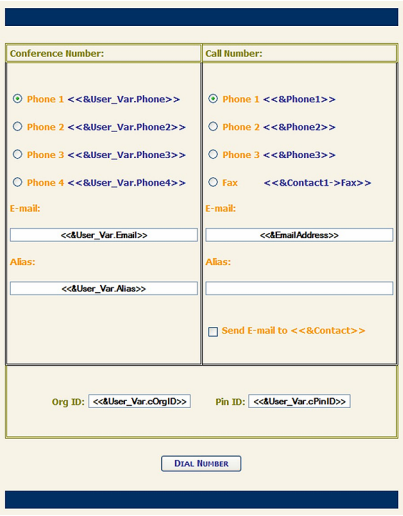


Figure 11-11

Linked Images

Are you a visual person? Are you a recruiter? Have you pictures that you would like to see of and for your candidates? For this next GM+View exercise, I will walk you through the simple creation of a GM+View template that will have some GoldMine fields, as well as a linked picture. I will do this directly in GoldMine, using the GoldMine tools available, just as an exercise for this book.

As I stated earlier, I would normally create my documents outside of GoldMine, and then incorporate them into the GM+View template. However, for you, the reader, I will work on an example directly within GoldMine.

I ask that you begin by creating a new GM+View document, and giving it a **Template Name:** of **Contacts Image**. Then tab to the body of the template.

The first thing that I will attempt to have you accomplish is to insert a picture place holder. I should explain this a little first. As this is a template, you can insert a place holder for a picture into the template. The place holder will include a variable by which the picture can be linked to a contact record under the GoldMine **Links** tab. When viewing the template under the GM+View tab, if there is a linked picture for that variable it will be displayed, else you will see a box with a red **X** in the upper left corner. See example in Figure 11-13 on the next page.

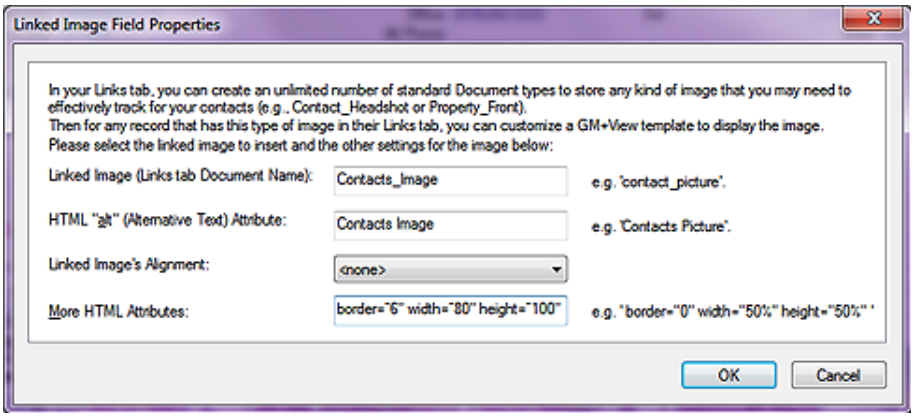


Figure 11-12

So then, let's begin by clicking the Insert field icon, refer to Figure 11-5. From the local menu select **Contact details ►** | **Linked Image** which will bring up the dialog shown above in Figure 11-12. You will notice that I have filled in some of the information in this dialog form already. For this GM+View Linked Image Field Properties dialog form, I intend the linked image to be a portrait of the contact. I

have, therefore, given the image the variable name of **Contacts_Image**. Additionally, I have included the alternate text of, **Contacts Image**, in case there is no linked image. The alternate text would be displayed identifying the place holder for the picture within GM+View, see Figure 11-13. I have chosen to do nothing about the place holder picture alignment as I just accept the default left location. I did include a border of **6**, a height of **100** and a width of **80** pixels. The height and width could have been entered as a percentage of the displayed page instead of using a fixed pixel height and width (see the examples to the right of each field in the dialog form). Actually, displaying examples is something that FrontRange is including in some of its newer dialog forms. There are examples right on the dialog form such that you should not need to select F1 for Help. When I click on the **OK** button, this is the string that I see in my GoldMine GM+View Editor:

<<LinkedImage:Contacts_Image:alt="Contacts Image" border="1" width="80" height="100">>

I could re-edit this information directly now to tweak it in, and I did, and/or I could add a new linked image by hand now that I know the syntax.

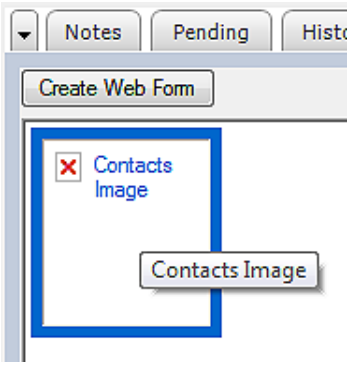


Figure 11-13

Close the **GM+View Tab Settings** dialog now so that you can put the place holder into action. I want to show you how to add a linked image to a specific contact record. Select the **GM+View** tab, and then select **Contacts Image** from the GM+Views drop list. Figure 11-13 displays what I see in my GoldMine, minus the cursor. Hopefully, yours is close to being the same. From here, I would ask you to move the cursor over the box. Notice that the cursor changes to a pointing fingered hand indicating that some action could take place with a simple click of the left mouse button. Click that mouse button now.

Do you recognize this dialog form, Figure 11-14? That's right, it's just the standard **Linked Document** dialog form with one exception, the **Document Name:** field has been populated for you with, in this case, **Contacts_Image**, and is uneditable. Click on the browse button (**...**), and browse to the image associated with your Contact that you want to have linked, a jpg file, I would hope. Now go ahead and click on the **OK** button. You can see that I used **C:\Inetpub\wwwroot\DJ Hunt\DWSite\Images\DJ_Cruise.jpg** as my personal image.

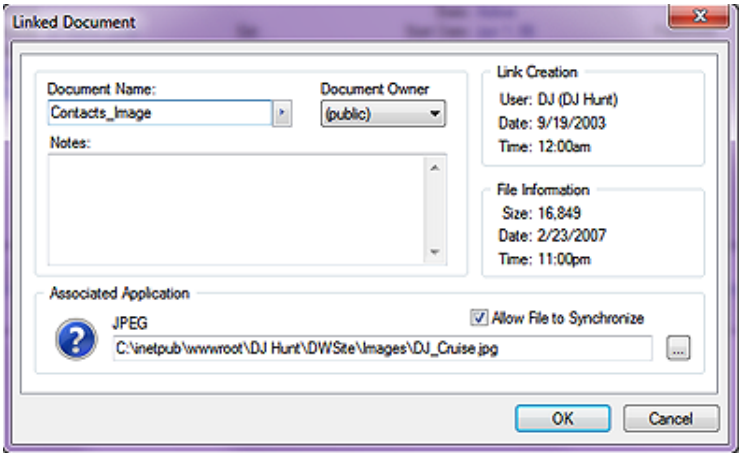


Figure 11-14

That's all there is to it. Now there is a Linked image for this contact record only, and, as important, the image will synchronize as long as you had selected to ☑ **Allow File to Synchronize** on the **Linked Document** dialog form. Figure 11-15 shows the results of this exercise based on my activity so far.

Figure 11-15

I feel that this is one of the really valuable bonuses represented by the addition of the GM+View to the GoldMine application. Imagine you were in Real Estate, and using the GoldMine Record Typing feature. You could have one Record Type for Clients, and another Record Type for Properties. On the properties record, you could be displaying images of the property, location maps, etcetera, easily, and quickly accessible for your prospective owners or your representatives to preview. Also, they are easily, and quickly accessible for e-mail to your client. Oh, the possibilities that the GM+View feature in GoldMine affords are virtually limitless. Add to this the GM+Browser, and you have a combination that cannot be beaten. When GM+View/GM+Browser was first introduced, FrontRange held a contest among its GoldMine Partners to design an award winning GM+View/GM+Browser. My Real Estate view, incorporating that which I discussed above, won second place, and was used as the GoldMine GM+View/GM+Browser Marketing brochure image for years.

Note

Actually, I'm going to tell you a little secret. After I created the screenshot for this book, I decided that my image looked weird so I modified the settings. They are actually:

border="6" width="90" height="100"

It was imperative that the text match the image, but I was just too skinny with the original settings.

Tip

If your organization will be synchronizing these linked images, I recommend that you create a folder under the GoldMine folder on the server, and then again on the remote installation alike. I then recommend that you place all of your linked images in this folder, and that the link to the image be made from this location.

Tip

Once you have the pointer HTML Code, in this case:

<<LinkedImage:Contacts_Image:alt="Contacts Image" border="6" width="90" height="100">>

...you could easily copy that into your hand built HTML document which might show 4 or 5 linked images along with formatted text as I discuss in the Properties GM+View.

You are limited only by your own design capabilities.

Let's go back in and edit the Contact Image a little more. This time I'll show you how to insert conditional text into the view. I have created, ahead of time, two text files, and saved them to my Temporary folder. They are **Fitchburg.txt**, and **Boston.txt**. Each text file contains some information relative to that particular city. I will use these in my example. You should create two similar text files to follow along or use the two that I have included in the accompanying template package.

Now, from the body of my template, I select the Insert field icon. This time I am selecting **Conditionally include a text file**. This causes the **Open** file dialog to pop up from which I select the first of my two files, Fitchburg.txt.

Once that is done, our template now contains:

```
<<LinkedImage:Contact_Picture:alt="Contact Picture" border="0">>

<<file: ?C:\Temporary\Fitchburg.txt>>
```

I position our cursor before the question mark (?), and enter my conditional expression such that the coding now looks like this:

```
<<LinkedImage:Contact_Picture:alt="Contact Picture" border="0">>

<<file: upper(&City) = [FITCHBURG]?C:\Temporary\Fitchburg.txt>>
```

Next I copy, and paste this statement again changing the conditional expression, and the text file that I want to use. The resulting coding now looks like:

```
<<LinkedImage:Contact_Picture:alt="Contact Picture" border="0">>

<<file: upper(&City) = [FITCHBURG]?C:\Temporary\Fitchburg.txt>>
<<file: upper(&City) = [BOSTON]?C:\Temporary\Boston.txt>>
```

I save this template, and close the **GM+View Tab Settings** dialog form. Now let's switch to the GM+View tab, if you're not already there, and look at the results.

Figure 11-16 shows my record for a person that resides in Fitchburg, Massachusetts. Notice that only one of the two conditional statements has been displayed, and, in fact, the correct txt file was displayed based on my conditional expression.

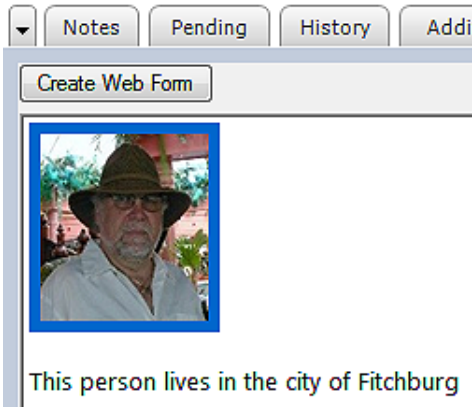


Figure 11-16

The possibilities are endless, and stretch as far as your imagination. Let's go back to a Real Estate example. You could have a text file with the property description, and have it display for the appropriate property.

How about security on a GM+View? Would you like to restrict a view to a certain user or users? That is just as easily accomplished using the conditional statement as I showed you previously, but with the condition modified. Something like this should do nicely:

```
<<file: &UserName $ [BG DJ RJ]?C:\GoldMine\HTMLTemplates\ThisHTMLFile.htm>>
```

Would you rather restrict it to a User Group? Something like this should serve you well:

```
<<file: &UserInGroup([Sales])?C:\GoldMine\HTMLTemplates\ThisHTMLFile.htm>>
```

Again, you are only bound by your imagination, and any bugs that you may come across in your experimentation.

Okay, let's continue on with one last item in our Contact Image template that I want to cover. I won't be covering the basic field item entries that are possible as you should have no trouble entering those. You should be testing those out on your own. I am going to enter a humongous expression, just to show you how this can be accomplished. This opens your GM+Views up to yet another set of possibilities.

I am going to use a rather extensive expression so I don't expect you to necessarily understand this expression, but you should have a general understanding of its results. I will explain that later. Again, I just want to show you that it can be done.

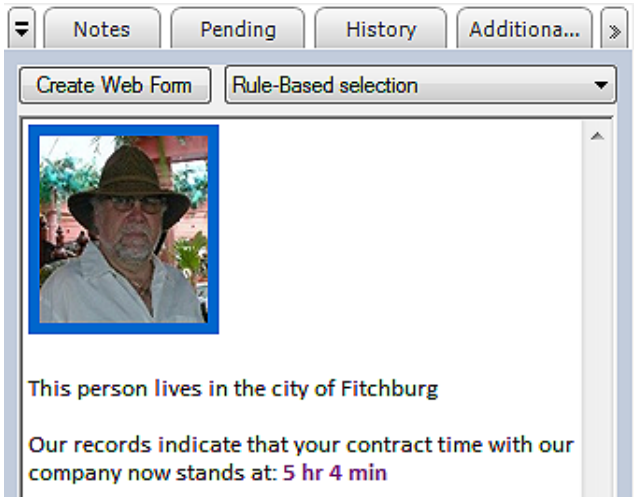


Figure 11-17

Here is the expression that I am entering at the bottom of the template:

Our records indicate that your contract time with our company now stands at: **<<&iif(Contact2.unCntTime>=0,alltrim(str(int(Contact2.unCntTime)))+" hr " +left(alltrim(str(60*(Contact2.unCntTime-(int(Contact2.unCntTime))))),2)+" min", iif(Contact2.unCntTime>-1,"-"+alltrim(str(int(Contact2.unCntTime)))+" hr " + substr(alltrim(str(60*(Contact2.unCntTime-(int(Contact2.unCntTime))))),2,2)+" min",alltrim(str(int(Contact2.unCntTime)))+" hr " + substr(alltrim(str(60*(Contact2.unCntTime-(int(Contact2.unCntTime))))),2,2)+" min"))>>**

Is that enough of an expression for you?

My GoldMine has a user defined field, **Contact2.unCntTime**, **N**, **10**, **2** in which I record the GoldMine Support Contract Time remaining on a customers contract. For this particular record, the contract time remaining is 5.07. I needed to convert that into the more readable hours and minutes format for my customers & users. The above expression comes right out of my GoldMine HTML E-mail Templates, however, it can be just as easily used in GM+Views. Figure 11-17 above, shows the resulting GM+View tab for this contact.

WOW! There is just so much that one can do with GM+Views that it just boggles the mind.

Note

You are reminded that this is one continuous expression, and that there are no linefeeds or carriage returns anywhere in this expression. You should be able to just copy & paste this into your example if you want to follow along.

External Table

I would like to now show you how to display any **External Table** within GoldMines GM+Views. This feature has been available since the GM+View/GM+Browser was added to GoldMine, hence, this section is applicable to all versions of GoldMine that contain the GM+View/GM+Browser feature.

Let's see if I can think of some place to use this feature. Hmmm! How about displaying data from GoldMine History for the active Contact record? Yes, I know that the History table is not truly an external table, however, as I plan to use it, it is in effect an external table. You will be able to use this same methodology to access any SQL table in the GM+Views/GM+Browser. I'm going to show you how to link back to the GoldMine SQL ContHist table, and to display formatted information from that table. As I use GoldMine Premium for Contracted Support (I am not yet using the Services module), I am going to use the ContHist table, and segregate out just those GoldMine history records that pertain to any Contract activity that has occurred for the active Contact record. I would remind you again, however, that you can connect to any external table(s) for which you have a bridge (ODBC or OLE DB). You should be able to take this discussion, and apply it to any external data source that you wish to access.

Internet Information Services (IIS)

To achieve my goal, I am going to have to make use of the **Active Server Pages** (**ASP**), hence, the need to have something on my server that will allow for the processing of asp files. My weapon of choice for internal asp processing is **Internet Information Services** (**IIS**). This automatically installs as part of your operating system, however, it is not usually configured automatically. This, then, may need to be done by hand.

Here are the steps to assure that you have this configured on, let's say, your GoldMine server. These steps are taken from my Windows 7 Ultimate system:

Step 1

In the Pearl Start Search area type in Program. You should see, appearing at the top of your list, Programs and Features. Click upon that option to bring forth the dialog form shown in Figure 11-18 on the next page.

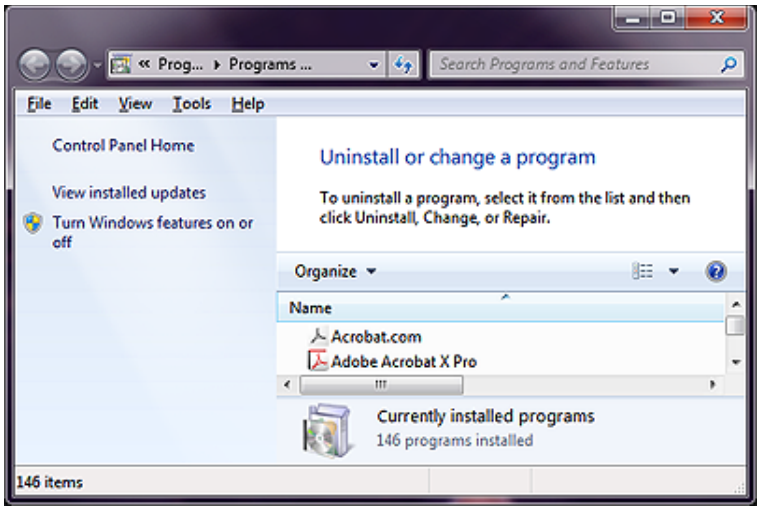


Figure 11-18

Step 2

Click on the option to **Turn Windows features on or off**.

Step 3

This should bring up the **Windows Features** dialog form shown here in Figure 11-19. Locate the checkbox for **Internet Information Services** (highlighted in Figure 11-19), and make certain that it is selected.

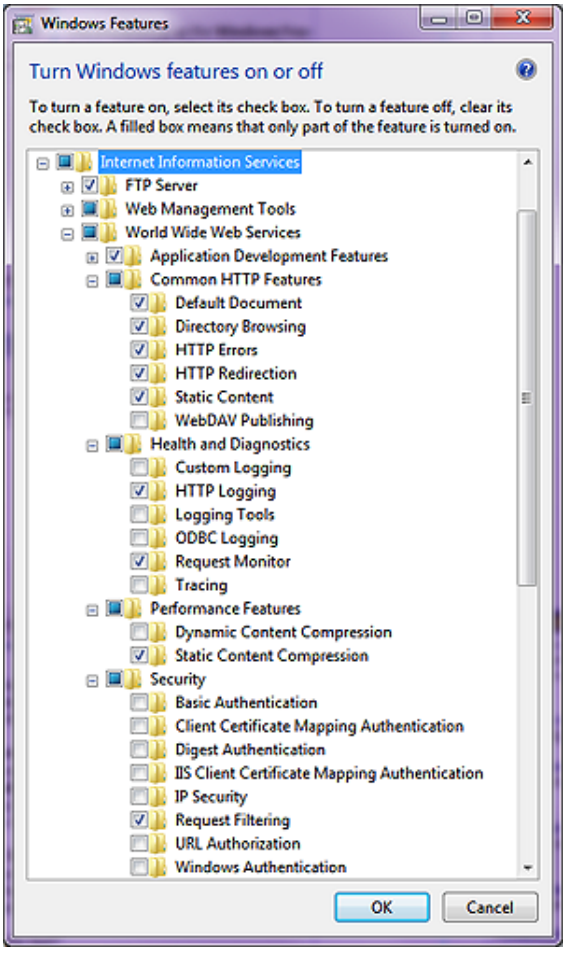


Figure 11-19

Step 4

If the box is checked, as shown here, then IIS has already been implemented for this system. On the other hand, if the option is not selected or partially selected, then place a check in the checkbox, and click the tree plus sign (**+**) to expand the tree as seen in the dialog form shown.

Here you would want to make certain that you have **World Wide Web Services** enabled. You may click upon the plus sign (**+**) to expand this tree as well. This should expose the tree further as seen in the dialog form displayed. In my configuration, not everything is checked. Naturally, your mileage may vary. Click on the **OK** button on this dialog form when finished.

After all that, click on the **OK** button, you may be asked to reinsert your **Windows DVD** so that the installer can capture some of the required components from the DVD. Alternatively, as in my case, the dialog form may simply close as I already have everything selected installed on my Window 7 Ultimate system. During the process you may be prompted to insert your Windows Installation disk, so be sure to have that handy when performing this action. Obviously, I have no idea which version your system will have installed. What I do know is that the build will be appropriate to your operating system, and that is what I care about.

Once the installation has finished, **Web Services** will be running automatically, and thereafter every time that you boot up your system. IIS installs mostly in the **C:\Windows\System32\inetsrv** folder, however, our interest is in the **C:\Inetpub\wwwroot** folder that is created at the same time. Under this folder I have created my working folder, **DJ_Asp_Book**, and would ask that you create a folder also as this is where I will ask you to put any work produced in this section.

As far as **Internet Information Services** (**IIS**) is concerned, that's all that there is to it. However, for safety, what say we test the service? Open your browser, and type in the address: **http://127.0.0.1/LocalStart.asp** or **http://LocalHost/LocalStart.asp** as either should work. Hopefully the page displayed correctly in your browser. You now have a web server, and are ready to proceed with your programming adventure.

Are you ready to put your programming skills to the test?

Scripting

Actually, it is not as hard as you may think. I will supply you with, and comment here on my **GM-ViewTable.asp** file. You will need a tool with which you can create your own script file. I use Adobe DreamWeaver, however, NotePad will work just as well. The only caveat that I would have for you is to not utilize Microsofts Word or Publisher for this effort. I have heard of too many cases where people have trouble with the resulting code from those applications. Remember, this is not a graphical page that we are creating, but script to send a graphical page of output to be displayed within, in our case, GM+Views.

Here is my script:

■ This section is required for the HTML document. Simply enter it as it is.

```
<%Option Explicit%>
<html>
<head>
<title>External Table in GM+Views</title>
</head>
```

■ This statement sets the background color for the GM+View displayed page, and you may change the color to match your schema

```
<body bgcolor="#ECE9D8">
```

■ And it begins for real. This marks the beginning of our script

```
<%
```

■ Here I am instantiating the variables that will be used throughout the script

```
Dim adOpenForwardOnly, adLockReadOnly, adCmdTable
Dim objConn, objRS
Dim strAccountNo, strSQL
```

■ Here I am presetting three constants to make them more readable in the script

```
adOpenForwardOnly = 0
adLockReadOnly = 1
adCmdTable = 2
```

■ Here I am creating two new objects to be used by the script

```
Set objConn = Server.CreateObject("ADODB.Connection")
Set objRS = Server.CreateObject("ADODB.Recordset")
```

■ Here I am populating the string AccountNo variable with the AccountNo as it was sent by GM+View in a QueryString. i.e.

...asp?AcctNo='A0101955535'&Company='Computerese'

```
strAccountNo = Request.QueryString("AcctNo")
```

■ Here I am actually making the connection to my SQL database. You will wish to set these properties specific to your SQL database.

```
objConn.Open "Provider=SQLOLEDB;" & _
             "Persist Security Info=False;" & _
             "Data Source=Server;" & _
             "Database=SQLGoldMine;" & _
             "User ID=sa;" & _
             "Password=SAPassword"
```

WARNING

*This section **must** be modified with the configuration for your system or the .asp will not function. Specifically:*

***Provider=SQLOLEDB** is for a Microsoft SQL backend.*

***Data Source=Server** requires the name of your SQL Server computer.*

***Database=SQLGoldMine** requires the name of your MSSQL database.*

***User ID=sa** is the System Administrator login for your SQL Server.*

***Password=SAPassword** where you would replace SAPassword with the password belonging to your System Administrators login.*

Note

*Pay attention the portion of the statement highlighted. I am only taking the first 11 significant characters of the **Contact1.AccountNo** field as my match. This avoids special characters that GoldMine uses, but at the cost of possibly retrieving more records than desired as this may mean that there is a better chance of duplication of AccountNos.*

*You may be able to get away with using the entire **Contact1.AccountNo** today. I haven't tried it, but I'm sure that you, my reader, will let me know if it doesn't function as expected.*

Note

Be careful of line wrapping for eBook presentation. Look at the original .asp file to see the proper syntax.

■ **Here I am defining a SQL Query, and placing it in a variable that I will execute against my connection string to my SQL database**

```
strSQL = "select Userid, convert(varchar,OnDate,7) as CallDate," & _
    "OnTime, " & _
    "Duration, " & _
    "Notes, " & _
    "left(Ref,charindex('(oc:',Ref)-1) as Reference, " & _
    "substring(Ref,charindex('oc:',Ref)+3, len(Ref)-charindex('oc:',Ref)-3) as Contact " & _
    "from ContHist " & _
    "where ActvCode like 'C%' and " & _
    "ResultCode = 'COM' and " & _
    "OnDate+30 >= getdate() and " & _
    "'''' & left(strAccountNo, 11) & ''' = left(ContHist.AccountNo, 11)" & _
    "order by OnDate desc"
```

■ **This is where the query is actually executed, and the results of the query is stored in the Record Set aliased as objRS**

```
Set objRS = objConn.Execute (strSQL)
```

■ **Here I begin to develop the HTML code that will be returned to the requestor, GM+View. I begin by defining a table.**

```
Response.Write "<P><H2><Font color='#000080'>QuickBooks History</Font></H2></P>"
Response.Write "<table id='myTable' cellspacing='0' border='0'>"
Response.Write "<tbody>"
Response.Write "<tr>"
```

■ **Here I add the first row, or header row to the table**

```
Response.Write "<H5><Font color='#000080'>"
Response.Write "Contract Time Remaining: " & Request.QueryString("ContractTime")
Response.Write "</Font></H5>"
Response.Write "<td><H5><u><font color='#000080'>User_ID</font></u></H5></td>" & _
               "<td><H5><u><font color='#000080'>Call_Date</font></u></H5></td>" & _
               "<td><H5><u><font color='#000080'>Duration</font></u></H5></td>" & _
               "<td><H5><u><font color='#000080'>Reference</font></u></H5></td>"
Response.Write "</tr>"
```

■ **Here I begin to populate my columns by looping through the returned Record Set. With each pass I am actually populating three rows of the table.**

```
If Not objRS.EOF then
    Do While Not objRS.EOF
```

■ **This is the first row in my GM+View, and it is populated with the UserID, Call Date, Duration, and the Reference.**

```
Response.Write "<tr>"
Response.Write "<td width=50><font size='2' font color='#000080'>" & objRS("Userid")
        & "</font></td>"
Response.Write "<td width=65><font size='2' font color='#000080'>" &
        objRS("CallDate") & "</font></td>"
Response.Write "<td width=50><font size='2' font color='#000080'>" &
        objRS("Duration") & "</font></td>"
Response.Write "<td><font size='2' font color='#000080'>" & objRS("Reference") &
        "</font></td>"
Response.Write "</tr>"
```

■ **This is the second row in my GM+View, and it is populated only with the Notes information.**

```
Response.Write "<tr>"
Response.Write "<td><font size='2' font color='#993333'>" & " " & "</font></td>"
Response.Write "<td><font size='2' font color='#993333'>" & " " & "</font></td>"
Response.Write "<td><font size='2' font color='#993333'>" & " " & "</font></td>"
Response.Write "<td><font size='2' font color='#993333'>" & objRS("Notes")
        & "</font></td>"
Response.Write "</tr>"
```

■ This is the third row in my GM+View, and it is populated with a period in the last column, in effect, creating a blank row between records.

```
Response.Write "<tr>"
Response.Write "<td><font size='2' font color='#993333'>" & " " & "</font></td>"
Response.Write "<td><font size='2' font color='#993333'>" & " " & "</font></td>"
Response.Write "<td><font size='2' font color='#993333'>" & " " & "</font></td>"
Response.Write "<td><font size='2' font color='#993333'>" & "." & "</font></td>"
Response.Write "</tr>"
objRS.MoveNext
Loop
End If
```

■ After I have finished looping through my record set to End of File, eof(), I have some programming chores to accomplish, like closing my objects in reverse of their creation.

```
objRS.Close
objConn.Close
```

■ Adding the closing markers for my table.

```
Response.Write ("</tbody>")
Response.Write ("</table>")
```

■ Setting the objects to nothing releases their memory allocation.

```
Set objRS = Nothing
set objConn = Nothing
```

■ And, lastly, my closing script marker, and my closing document markers.

```
%>
</body>
</html>
```

WARNING

*Notice, in the **Contract Usage.html** document, that the path is to my server, and the location of the .asp which resides on my WebServer. This path must be modified in your document prior to employing it in your GM+View.*

This is but an explanation of the **GMViewTable.asp** document which is called by the **Contract Usage.html** document, both of which I have supplied to you in a zip file included within the download version of this eBook. You could just drop that script into your **C:\Inetpub\wwwroot\Asp_Book** folder, modify the connection string to match your system, and it should work with the proper call from GoldMine GM+Views. Review the sidebar **WARNING** on this.

Next we must have a GM+View template that calls our .asp, see included **Contract Usage.html**. Remember that I advocate creating your templates outside of GoldMine, and then using the GM+View **Include a text file** option in the body of the template.

My template code is very simple, but the statements are very long, and wrap for eBook presentation. Please review the actual document supplied while I am covering it here.

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/
xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
    <head>

        <!--
            This GM+View Template is provided as part of GoldMine Premium - The Definitive Guide
            series of books, and the author is providing this without guarantee or additional support.
        -->

    <meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1" />

    <title>Contract Usage Document</title>

    <Script language=JavaScript>

        <!--
        function redirect()
        {
```

■ These next 3 lines are wrapped, but in the document the line is one continuous line without spaces.

```
        window.location = "http://DJ-Workstation/DJ_Asp_Book/GMViewTable.asp?AcctNo=&lt;
                &lt;&amp;left(AccountNo, 11)&gt;&gt;&Company=&lt;&lt;&amp;Company
                &gt;&gt;&ContractTime=&lt;&lt;contact2->unCntTime&gt;&gt;"
```

```
        }
        setTimeout ("redirect();", 0000)
        // -->

    </Script>

</head>

<body>
</body>

</html>
```

Once you have modified **Contract Usage.html** to your specifications, you may include it in the body of your GM+View template as I have done for the many other templates that I have discussed in this chapter.

Well now the acid test. Let's try it out. Selecting this view from the GM+View should produce a view similar to that shown here in Figure 11-20.

Figure 11-20

Does yours look similar to Figure 11-20? My guess would be not. You may have noticed that my SQL Query was looking for a specific type of Activity Code having a specific type of Result Code, neither of which may be applicable to your situation. This brings up a good point. You must make your SQL Query draw the data that you want to display. Try your SQL Queries out first in GoldMine to make certain that they are pulling the data that you expect to display in GM+Views, and then modify your SQL Query in your html document appropriately.

Let me explain a little about what this query is filtered against. My structure makes pulling out the where clause statements easy, as each is on its own line.

- **Where the ContHist.ActvCode begins with C**

```
"where ActvCode like 'C%' and " & _
```

- **Where the ContHist.ResultCode equals COM**

```
"ResultCode = 'COM' and " & _
```

- **Where the ContHist.OnDate is not older than 30 days ago**

```
"OnDate+30 >= getdate() and " & _
```

- **Where the first 11 characters of the ContHist.AccountNo equals the sent value**

```
"'" & left(strAccountNo, 11) & "' = left(ContHist.AccountNo, 11)" & _
```

With this as the basis of your work, you should be able to pull data from any source into your GM+View for which you have an **ODBC** or **OLE DB** driver. There are endless possibilities, and I hope that you will let your imagination run wild. If you do create some nice GM+Views that you would like to share with my audience, please send them along to DJ@DJHunt.US.

contributing author:
Andrea Dominguez

12

GoldMine Report Writer

In This Chapter

This is the chapter that has been years in the making. Not knowing enough myself about the GoldMine Report Writer, I have asked Andrea Dominguez, my go to reports person, to write this chapter for us. I must say that she has done an exemplary job.

This chapter is to discuss the navigation, alteration and creation of reports using the GoldMine Report Writer.

Several already created reports come with your copy of GoldMine Premium as delivered. If you only needed to print reports without altering dates or users, you can go to File | Print a Report.... All reports that are pre-made reside under the UserID of (public). To make alterations to reports, change parameters or create new reports select from the GoldMine menu:

Go To
Reports ►
Reports

From this dialog form, shown later in this chapter, reports can be cloned or altered by adding fields and/or parameters called dialogs. One thing these reports cannot do is be saved as an Excel spreadsheet or a PDF, although they can be printed to a PDF printer. GoldMine Reports can be saved as an **.rtf**, which can be opened in Word, or in the GoldMine Report Writer's proprietary format of **.frc**.

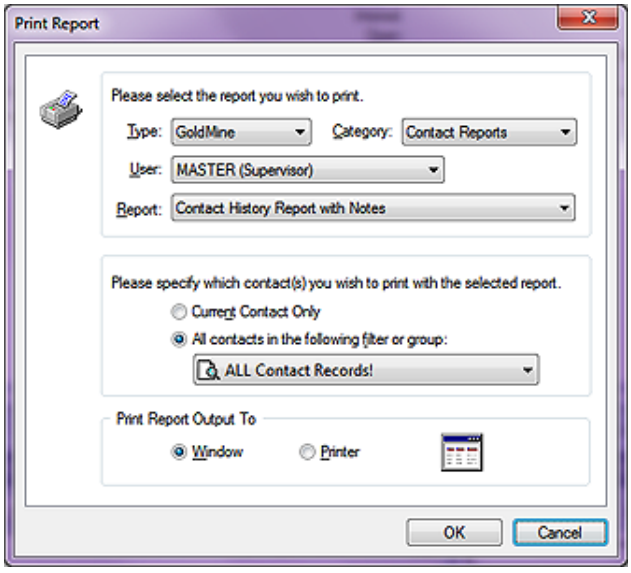


Figure 12-1

Let's first bring up the dialog form, displayed above in Figure 12-1, by selecting from the GoldMine menu:

File
Print a Report

In this dialog form we have three different frames, two untitled, and one titled. Let's begin by looking at the top most untitled frame in which there is a statement: **Please select the report you wish to print.**. The first entity in there is **Type:**. If you have Crystal Reports installed, and those crystal files are in your GoldMine folder, then there will two choices in this drop down, **GoldMine** or **Crystal** otherwise only GoldMine will be available for the **Type:** of report. This tells GoldMine what drivers should be loaded to run the report.

The **Category:** drop list is the next item of interest in this frame. There are 6 different categories from which to choose. These categories are available for

Printing a Report

Tip

As a general Rule of Thumb, do not place Calendar based reports in the Contact record category, nor place Contact record listing reports in another category.

Tip

*If you decide to customize one of the reports that come with GoldMine for your users, it is recommended that you **Clone** the reports under a **UserID** name instead of the user **(public)**. This makes the customized reports easier to find and accessible by all.*

Tip

*Utilizing **Filters & Groups** for record selection will make the report run faster, and it is recommended to use the **Filters & Groups** externally rather than have the report do all of the filtering at runtime.*

Tip

Make certain that you have defined your Printer prior to running any report that you intend to print.

Tools
Configure ►
Printer Setup...

Note

If you receive an error message while trying to send your report to the printer that states printer cannot exceed 59 characters, then change the name of the printer to which you are sending the print job. The path and/or name is too long, and GoldMine has a limit on reading paths.

Tip

*For large reports without a **Notes** field, the performance will be better if you first print it to the window, then send it to the printer.*

Tip

While troubleshooting why reports are not being sent to the printer from a Window view, try sending the report directly to the printer. There are some environments that sending directly to the printer works, when the Window view does not.

The Report Center

Note

The highlighted area, Figure 12-2, displays how many reports are contained within each category.

both GoldMine reports, and Crystal Reports. You should familiarize yourself with the various reports contained in each category prior to attempting to print a report.

- **Contact Reports** - these are reports that are based on the Contact record such as address lists. While calendar reports or history reports, in general, are not stored in this category, this category does contain some Pending and History reports.
- **Calendar Printouts** - this category includes graphical calendar reports as well as reports that will fit in Daytimer binders.
- **Service Reports** - this category is relatively new to GoldMine Premium. It includes a few default reports regarding Cases (Opened and Resolved), as well as a couple of detailed reports from the same category.
- **Analysis Reports** - this category contains Opportunity Manager reports as well as various Forecasted Sales Analysis reports.
- **Labels and Envelopes** - this is a self explanatory category, and contains various Avery Label reports, and Monarch Envelope reports.
- **Other Reports** - these reports are odds and ends reports in the GoldMine application such as the Filters Listing & Rolodex Reports.

As you change the category, the report selection changes accordingly.

All of the reports that come pre installed with GoldMine are located under the **User: (public)**.

Lastly, in this frame, is the **Report:** that you wish to have printed. Only those reports that belong to the specified **User:**, that are in the specified **Category:** will appear in this drop list.

The next untitled frame, as identified by the statement: **Please specify which contact(s) you wish to print with the selected report.**, is to tell the report which GoldMine records to include in the report.

- **Current Contact Only** - will run the report only against the information from the currently active Contact record.
- **All Contacts in the following filter or group:** - the default of which is **ALL Contact Records!**, however, this option will select the records based on any already created Filter or Group located under the current User Filter/Group menu or that of any User Filter/Group in the GoldMine system to which the logged in UserID has access rights.

As these are a radio button selection one may only select the one or the other, and not both.

In the final frame, **Print Report Output To**, you have a radio button selection option as well. You may either select ◉ **Window** or ○ **Printer**. Output to Window will print the report to the screen as the output to Printer sends the report directly to the preselected printer (see sidebar Tip).

Finally, you may click upon the **OK** button to begin the report processing.

As mentioned before, all installed reports reside under the **UserID** of **(public)**. All users have access to this user even under the strictest of security setups unless you remove complete access to the Report Center. Sometimes it gets confusing finding custom made reports in the sea of pre-made reports under the **UserID (public)** unless you have a very good naming scheme in place. Some scenarios of this nature would be like this, either create a new GoldMine **UserID** named **Reports** or save all the reports under the **UserID MASTER** while giving the other users access to these reports. All users would then be able to view the custom made reports easily, and use them as well.

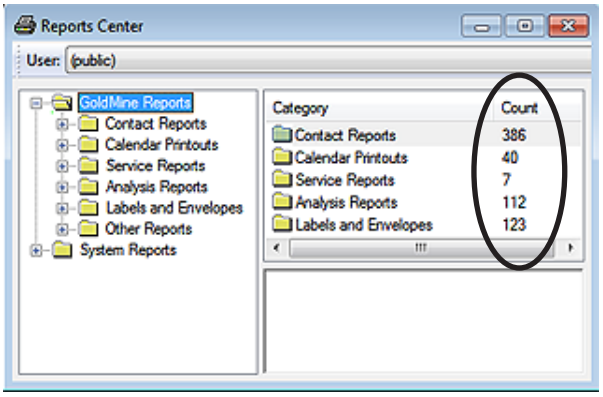


Figure 12-2

Let's bring up the **Report Center** by selecting from the GoldMine menu:

Go To
Reports ►
Reports

You will notice, at the top of the reports menu, there is a drop down list for **User:**. When you login to GoldMine for the first time, the list item defaults to the user **(public)**.

In the frame immediately below the **User:** drop down list in the tree there are 3 possible report type choices:

GoldMine Reports
Crystal Reports (Type only present if Crystal RunTime files are in the GoldMine root directory)
System Reports

Under the GoldMine and Crystal sections, there are subsections called categories. These categories cannot be edited, nor can other categories be added. It is recommended to be mindful of where you are saving your cloned or newly created reports in regards to these categories. You should not place a Calendar report under the Contacts or Labels category.

Note

You may have noticed, Figure 12-3, that I have exposed the Report Center Toolbar. I have done this by dragging the highlighted handle of the Report Center Toolbar under the ***User:*** *Toolbar.*

Note

Highlighting a ***Category*** *from the tree exposes a list of the reports contained within that category to the right. Highlighting a* ***Report*** *from the exposed list will cause an image of the report to display below the list items. This only holds true for the default GoldMine reports unless you create your own report images in the proper manner for the reports which you create.*

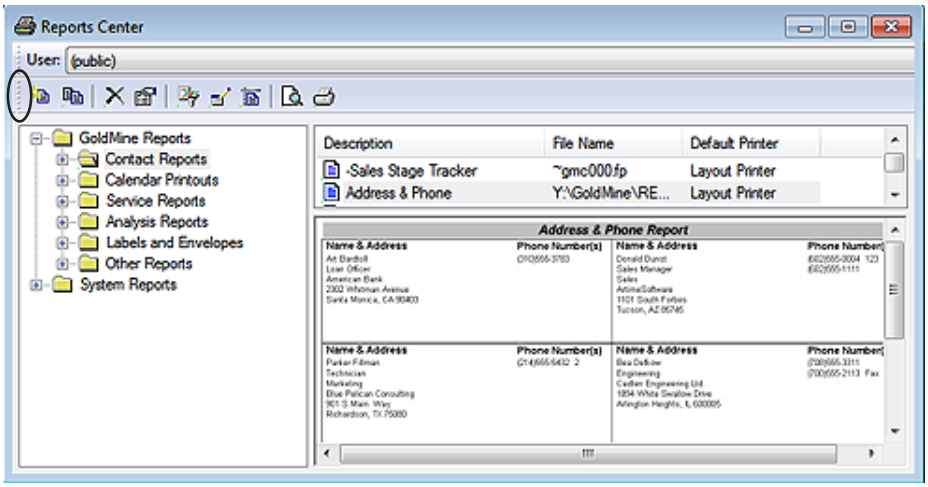


Figure 12-3

Note

For the report image to display, the image must reside in the ***Reports*** *folder under the* ***GoldMine*** *folder, and it must have the exact same name as the report with the* ***.jpg*** *extension.*

Let's start with running an existing report. Refer to Figure 12-4 below. Expand the subcategory branch called **Contact Reports** located under **GoldMine Reports** tree in the left column for the **(public)** UserID. You should see several preexisting reports in this column. By clicking once to highlight a report in the tree the right frame of the Report Center dialog form comes to life. You should also see a **jepg** of the report in the bottom right frame as well as a **Description**, the **File Name**, and the **Default Printer** it is designed to run with in the top right frame. Layout printer means that the print driver that the report will use is that of the default printer selected in GoldMine at designed time. Selected printer is the printer the user chooses. This does make a difference at runtime. You might find that some things that run fine for you might be chopped off for another user. This is because of the various possible printer drivers called at runtime.

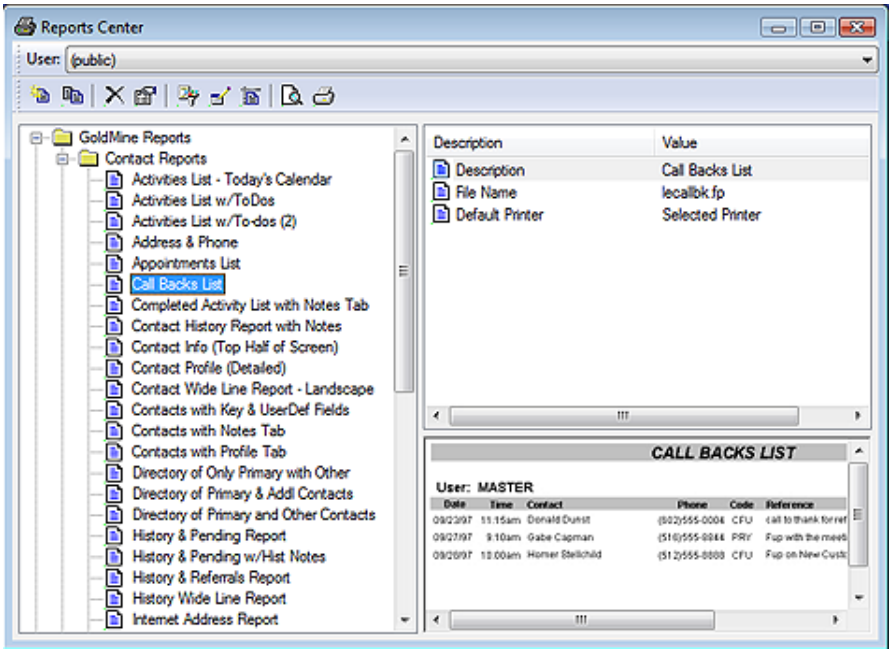


Figure 12-4

Existing Reports

Contact Reports

Note

Remember, in GoldMine Premium, each default report has an associated example image of that report.

*For the report image to display, the image must reside in the **Reports** folder under the **GoldMine** folder, and it must have the exact same name as the report with the **.jpg** extension.*

When the report is highlighted, in the list of reports, the report example will be displayed in the lower right quadrant of the dialog form.

Your copy of GoldMine comes with pre-made reports. These reports are located under the UserID **(public)**, and the report type of GoldMine. By right clicking on a report, and left clicking on **Properties...** you can find additional notes on how the report runs as well as for what it may be used.

Under the **Contact Reports** category branch, the following reports should be available;

Description	File Name	Report Summary
Activities List - Today's Calendar	abactvt.fp	Reads from System date, and returns only activities for run date.
Activities List w/ToDos	abactvu.fp	Works with Options tab, and returns activities based on selection.
Activities List w/To-dos (2)	abactlst.fp	Works with Options tab, and returns activities based on selection.
Address & Phone	leaddrph.fp	Works with external filters, and gives summary in business card size area.
Completed Activity List w/Notes	lecmphst.fp	Displays completed activities by contact record as well as the Notes tab. Works best with current contact or filtered group - should not be used on all contact records unless database is very small.
Contact Details	contprof.fp	Displays a snapshot of complete record, Pending, History, Details. Works best with current contact.
Contact History Report w/Notes	lecomhst.fp	Displays Contact1 address info as well as completed activities, and the Notes entered on those activities.
Contact Info (Top Half of Screen)	smpcontp.fp	Displays Contact1 info as well as Summary tab, and User Defined fields.
Contact Wide Line Report-Landscape	contline.fp	Displays Address information is landscape format. One record per line. Works best for large filters.
Contacts with Details Tab	smppt.fp	Displays Address info, UserDef fields, Key fields, Summary tab, and Details tab.
Contacts w/Key & UserDef Fields	lekeyusr.fp	Displays Company, Primary Contact, UserDef fields, Key fields, and Notes.
Contacts w/Notes Tab	smpnt.fp	Similar to Contact Profile Detailed except only displays Contact1, Contact2, and Notes.
Directory of Only Primary w/Other	wwdir500.fp	Displays Primary Contact as well as Other Contacts. Will not display if nothing is entered under the Contacts tab.
Directory Prim & Addl Contacts	abpcoc.fp	Same graphical format as above listed report, list all records in external filter criteria.
Directory Prim & Other Contacts	wwdir.fp	Same basic graphical format, and properties as above listed report. Other Contacts display in bold.
History & Pending Report	smph&pts.fp	Displays Address info, Key fields, User Defined fields, Pending & History activities.
History & Pending w/Hist Notes	abp&hn.fp	Displays same as above listed report as well as Notes entered in History activity.
History & Referrals Report	histref.fp	Displays Address, Key fields, User Defined fields, History activity - no Notes. Good for current Contact only or small filter.
History Wide Line Report	histline.fp	Landscape format, lists 1 history activity per line. Good for running against larger filters, date range set in Options tab.
Internet Address Report	Intadd.fp	E-mail Address list. Good for small filtered groups due to format. Not meant for current Contact.
Linked Documents	lelink.fp	Displays Linked Documents. Best used in filter or all Contact records. Not meant for current Contact only.
Message List	smpml2.fp	E-mail sent report. Not meant to use on current Contact only. Date range set in Options tab.
Next Actions List	smpnal2.fp	Displays Pending Next Actions. Date range, and User set in Options tab.
Organizational Tree	orgtree.fp	Lists all Relationship trees. MUST use all contact records at runtime. Will not work with filter or current Contact.
Other Contacts Listing	oclist.fp	Lists Additional Contacts. Not meant for current Contact only.
Pending Activity Report	lefutact.fp	Displays Pending activities specified in Options tab.
Pending Call List	lecallbk.fp	Returns calls in range selected in Options tab. Even though it says Call Backs, it means all calls.

Pending Wide Line Report	smppwlr.fp	Landscape format. Displays Pending activities specified in Options tab.
Phone Book	fonelist.fp	Contact1 telephone list. Can use Sorts tab to set lookup or external filter or both.
Phone Book (Primary & Additional)	foneoth.fp	Phone list for both Primary and Additional Contacts. Can use Sorts tab, and external filters.
Phone Book Business Card Style	fonebus.fp	Lists Primary Contact record Phone, Address info in business card graphical format.
Phone1 & Fax Report-Landscape	le4fones.fp	Lists Company, Contact, Phone1 and Fax.
Potential Dupe Contact Report	smpdupch.fp	Not the most accurate of reports, looks for potential duplicate records.
Referral List	abreferl.fp	Displays Contact1 info for Referrals made; this record was referred to this record.
Referral Status Report	refstdtl.fp	Lists Contact1 info for Referrals as well as the status selected in Referral tab properties.
Today's Appointments	leapptls.fp	Returns appointments selected in Options tab.
To-Do Listing	smptd.fp	Lists To-Do's. MUST use all Contact records. Will not work with current Contact or filter.

Calendar Printouts

Under the **Calendar Printouts** category branch, the following reports are available:

Description	File Name	Report Summary
*Calendar Default Report	c1.fp	Lists Calendar Notes, not activities.
Activities Monthly	lemonth.fp	Displays Calendar in Month graphical format.
Daily - DayTimer (detailed)	caldv.fp	Displays by Day, Daytimer format Calendar with Contact details. Folio size only.
Daily - DayTimer (summary)	caldf.fp	Displays by Day, Daytimer Summary format. Folio size only. By default will insert current date into date range, but you can change that.
Daily Appointment List	c1fp000.fp	Displays Scheduled Appointments, Contact, Duration, Time, Date, Reference, and Notes.
Monthly - DayTimer	calmv.fp	Displays Calendar in Month graphical format. Folio size only, and runs current month but you can alter dates.
Weekly - Day Timer*Avery 41357	calwl.fp	Prints 2 Daytime Desk size pages per 8 1/2 X 11 piece of paper.
Weekly - DayTimer (portrait)	calwl000.fp	Prints Folio size Daytimer calendar in Week graphical format. Auto inserts current week, but dates can be altered.
Weekly Appointment List (portrait)	smpwalst.fp	Prints Folio size Daytimer in Week graphical format. Only displays Appointments.

Note

Remember, in GoldMine Premium, each default report has an associated example image of that report.

*For the report image to display, the image must reside in the **Reports** folder under the **GoldMine** folder, and it must have the exact same name as the report with the **.jpg** extension.*

When the report is highlighted, in the list of reports, the report example will be displayed in the lower right quadrant of the dialog form.

Note

*If you wish to alter the dates that the report is looking at, you will not be able to reach the **Options** tab from the **Print a Report** menu. You will need to go to the **Report Center**, and open the **Options** tab.*

WARNING

*One should **never** consider modifying any of the default reports in any manner. One should always **Clone** the report, and then modify the **Clone**.*

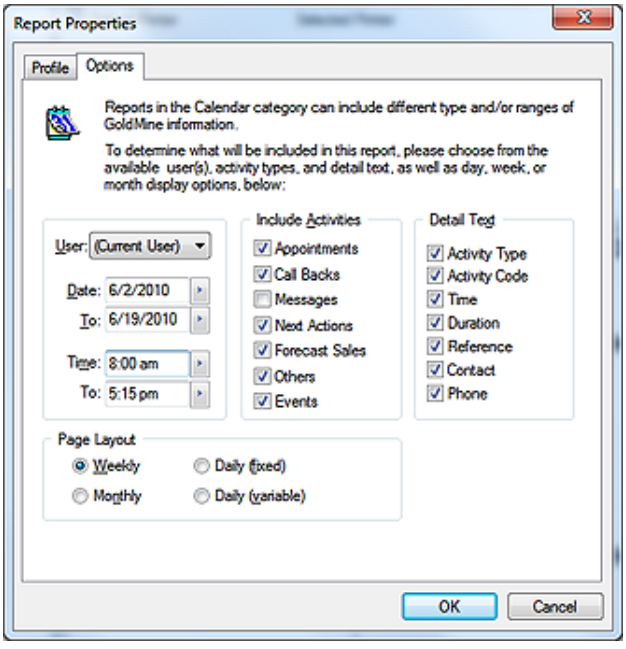


Figure 12-5

Why don't we take a look at the **Options** that I have been discussing. You can bring up Figure 12-5 in your copy of GoldMine Premium by right-clicking on any report within the **Report Center**, and selecting **Properties...** from the local menu. Next click on the **Options** tab in the dialog form. I might add that the **Options** tab displayed in Figure 12-5 is from a **Clone** of one of the **Weekly Day Timers**, which brings up a good point, never work on the original, always work on a Clone.

You will notice that there are four frames on this dialog form this time, one untitled, and three that are titled. The first frame, and also the untitled frame, is the first of our filtering sets. You may set the **User:** to filter the report for the **(Current User)**, the user that is logged into GoldMine, or

alternatively, for a specific user. For these particular report types, you cannot Filter against a User Group. **(Current User)** will display only the activities for the person currently running the report, and is typically the best choice. In the same frame, we next have a date range that can be applied as our Filter. There is the from **Date:**, and the **To:** date for the Filter range. Lastly, and still in this frame, we also have a time range that can be applied. There is the from **Time:**, as well as the **To:** time for the completion of the range.

The next Filter that we can apply is via the **Include Activities** frame. As the frame title indicates, you may Filter on the various **Cal.RecTypes** (refer to The Tables chapter) that are available simply by checking or unchecking the appropriate boxes. Those that are checked will be included in the report at runtime, while those that are unchecked, at runtime, will not be included.

Now it is your choice as to the information to include in these types of report, and you do so using the **Detail Text** frame. Again, you have checkbox options to toggle on or off. I won't reiterate the list here, however, you have 7 pieces of additional information that may be included in your output at runtime.

Lastly, on this dialog form, you have the **Page Layout**. Notice that there is no hot key for this frame. As a general rule, do not change the settings in this frame.

Service Reports

And under the **Service Reports** category branch, the following reports are available:

Description	File Name	Report Summary
Assigned Cases by User	SRAssignedCases.fp	Prints Assigned Cases by the UserID, Start Date.
Resolved Cases by User	SRResolvedCases.fp	Prints Resolved Cases by the UserID, Start Date.
Resolved Cases by User per Mont	SRResolvedCasesPer Month.fp	Prints Resolved Cases by the UserID, Start Date with a further breakdown by Month.
Service Report - Pending & History	SRH&P.fp	Prints the Case Details as well as a listing of any Pending or Historical activities related to this Case.
Service Report w/Tasks	SRT.fp	Prints the Case Details as well as a listing of open Tasks remaining for this Case..

Analysis Reports

Under the **Analysis Reports** category branch, the following reports are available:

Description	File Name	Report Summary
*Opportunity Code Report	abopidrp.fp	Prints Opportunities by the ID GoldMine assigns, UserID, Start date.
*Opportunity Report - Pending & History	aboph&P.fp	Prints full detail of opportunity. Pending items as well as History, Contact information.
*Opportunity Report w/Tasks	abopmgrr.fp	Prints Opportunity with Task details, Competitor, Influencers, and Contact information.
*Project Code Report	oppcdrpt.fp	Prints Projects by the ID GoldMine assigns, UserID, Start date.
*Project Report - Pending & History	abpjh&p.fp	Prints full detail of project. Pending items as well as History, Contact information.
*Project Report w/Tasks	abprjmgr.fp	Prints Project with Task details, Competitor, Influencers, and Contact information.
Completed Sales by User	compsale.fp	Prints Completed Sales for all Users. This can be changed to a single user in Sorts tab.
Completed Sales by User per Month	smpcsum.fp	Prints Completed Sales for Current User, and lists by Month Sale in which the sale was completed.
Completed Sales by User per Month (2)	conta003.fp	Prints Completed Sales for all Users, and lists by Month Sale in which the sale was completed.
Forcasted Sales Funnel by Cycle(detail)	smpfsfcd.fp	This report has a dialog for rate range, do not use the Options tab to set the date range. The Funnel is a display Summary by Probability.
Forecasted Sales by User	smpfsru.fp	Lists Forecasted Sales for Current User as well as a total for the Company, and Closing Ratio based on Probability.
Forecasted Sales by User (2)	smpsfse.fp	Lists Forecasted Sales for Current User.
Forecasted Sales-User per Month	smpfsum.fp	Prints Forecasted Sales for Current User, and lists by Month Sale according to sale date.
Forecasted Sales by User per Month (2)	smpsfum.fp	Prints Forecasted Sales for all Users, and lists by Month Sales according to sale date.

Note

Remember, in GoldMine Premium, each default report has an associated example image of that report.

*For the report image to display, the image must reside in the **Reports** folder under the **GoldMine** folder, and it must have the exact same name as the report with the **.jpg** extension.*

When the report is highlighted, in the list of reports, the report example will be displayed in the lower right quadrant of the dialog form.

Note

By program default, these reports insert a date range that includes anything that you'll ever do. To help the report performance, change the date range to only what is actually needed.

Note

*For reports that use a dialog for the date range, make sure your **Options** tab for that report has an end date that is very far into the future (by as much as 10 years or more).*

Forecasted Sales Funnel/Month(Yearly)	smpfsfms.fp	Prints Forecasted Sales. Uses dialog for date range. Also displays numeric information by Probability.
Forecasted Sales Funnel by User	smpsffl.fp	Same graphical display as Forecasted Sales Yearly, but by User. Also uses dialogs for date range.
Forecasted Sales Funnel-User (Detail)	smpsffl2.fp	This report has a dialog for date range, do not use Options tab to set the date range. It also lists for the Current User. The Funnel is a display summary by Probability.
Opportunity Manager: Opportunities	smpo&p.fp	Prints existing Opportunities. Can also use on the current selected record.
Opportunity Manager: Projects	smpo&p2.fp	Prints existing Projects. Can also use on the current selected record.
Opportunity Manager: Opportunities (2)	abopmgro.fp	Prints existing Opportunities, but advances page after each one.
Opportunity Manager: Projects (2)	abopmgrp.fp	Prints existing Projects, but advances page after each one.
Phone Calls Log	conta001.fp	Prints on Current User. Lists Phone Calls com-pleted as well as at end of report gives Statistics, Call made, Incoming, Outgoing, etc…
Phone Calls Statistics	conta002.fp	Gives Completed Phone Call Statistics, Duration, Incoming, Outgoing, etc…
Scheduled Activities Analysis	smpactan.fp	Gives Statistics to Pending activities by User. Total # Calls, Appointments, etc.… Can limit User in Sorts options.

Since I have discussed the possible use of the sorting option, it would only be fair that we discuss this in more detail. I have selected the **Sort Orders** option for the **Scheduled Activities Analysis** report as my example, and this can be seen here in Figure 12-6.

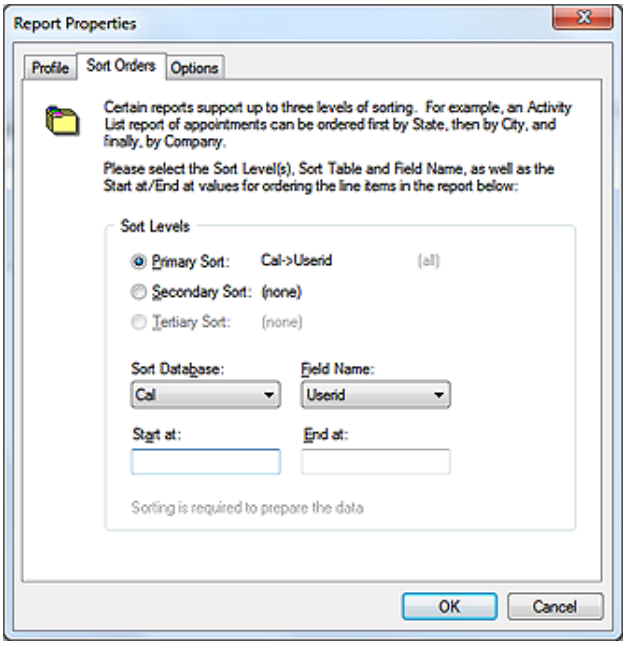


Figure 12-6

One of the first things that you may have noticed is that these are radio button options which would leave one to believe that this report could only have one sort order per report. Obviously, this could not be true, or very useful either. Again, the GoldMine developers have used a control, the radio button control, in a non conventional manner. In fact, there are three sort orders that you can control, the **Primary Sort:**, **Secondary Sort:** and the **Tertiary Sort**.

The ◉ **Primary Sort:** is always selected when this dialog form is viewed, and the information in the **Sort Database:**, the **Field Name:**, the **Start at:**, and the **End at:** fields all pertain to the selected sort which, in this case, is the primary sort.

You may then select the ○ **Secondary Sort:** option, and modify the information as is appropriate for your report.

Lastly, if you need it, you may then select the ○ **Tertiary Sort:** option, and customize the information for the tertiary sort as is appropriate to your report.

I would like to reiterate once more, that making any changes, any where on the delivered GoldMine reports could be detrimental. You should always, repeat always, be modifying a cloned copy of the report in case any changes that you make in your testing crash the report or make it unusable. Trust me as I speak from having worked to recover many client reports that magically just stopped working.

Note

*You could limit a report, that lists all Users, by changing the **Start at:** value which may have **AAA**, to the **UserID**, and then entering the same **UserID** in the **End at:** field.*

*So instead of it saying from **AAA** to **ZZZ**, it could now say from **ANDREA** to **ANDREAz**.*

Labels & Envelopes

Labels and Envelopes have pretty much had the same standard list since GoldMine 3.2 (perhaps even earlier). There are several different Avery label sizes, and standard size envelopes. Where the envelope size may have not altered since 1995, the labels have changed in size. They have been

changed to accommodate deskjets, inkjets and laser printers. Since there are so many printers on the market, the labels tend to creep at runtime. Later in the chapter I will give detailed instructions on how to prevent the label creeping.

Other Reports

Under the **Other Reports** category branch, the following reports are available:

Description	File Name	Report Summary
Filters Listing	wwfilter.fp	Lists filters for all UserIDs complete with the expression (in English as well as dBase). Can limit to single UserID in Sorts tab.
Lookup File Listing	lookup.fp	Lists the contents of a Lookup menu which is controlled in Sorts tab.
MailBox Report	smpmailb.fp	Displays E-mail sent and received. UserID and Date Range is controlled in Sorts tab.
Merge Form List	formlist.fp	Lists Merge Forms by UserID.
Scripts Report	lescript.fp	Displays Telemarketing Scripts
Users List	smpuslt.fp	GoldMine User list. Also shows last Login/Logout time, Date, Full username.

Customizing Reports

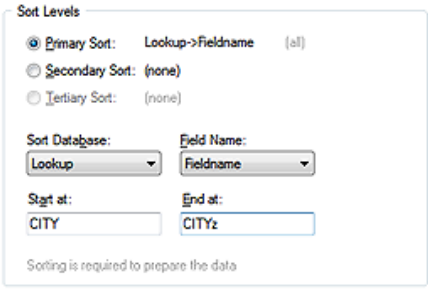


Figure 12-7a

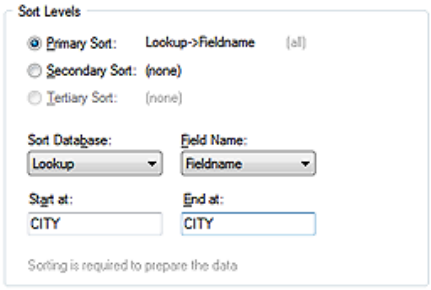


Figure 12-7b

Sort Levels

Pay close attention to **Sort Levels** in the **Other Reports** category branch, for that matter, pay close attention to **Sort Levels** in all category branches. For example, the **Lookup Listing Report** has, in the default state, on the **Lookup.FieldName** with no sort range specified. In Figures 7a & 7b above I show you two possible sort ranges. The sort range in Figure 7a, when printed, will produce results while the sort range in Figure 7b will produce a blank page. The first sort range, Figure 12-7a, requires an extra character to see the entire list, and the report returns what is asked for as expected. The second sort range, Figure 12-7b, is missing that extra character, and did not return any results.

You can have up to three sorts for any given report. Sorting is a type filtering that is activated at runtime. It does not function until all of the records have been read into the view by the report. **Sort Levels** is also how the report will be graphically grouped. When creating your report sorting, it is best to approach it as you would with creating a complex GoldMine Filter. The Primary Sort is the largest body of data to be reviewed. If you are creating a report on a contact listing by City, then your Primary Sort would be City unless the database is using record curtaining, and there are far more records in the curtained group. I showed you a typical **Sort Orders** dialog form in Figures 7a & 7b.

In the report itself, however, you could add a 4th sort. By going into the properties of the fourth sort, and adding an additional filter. You can also add further filters to the properties of all the sorts such as particular RecTypes.

Options

You can select Users, Date Ranges and Activity Types if you are running History or Pending reports. It is generally a good rule of thumb practice to limit user's exposure to this area. If the report needs to be flexible for items such as being able to change activity type or user, inserting runtime parameters, called **Dialogs**, is a much better approach.

Note

*If you are running a report such as **History & Pending Report** for a single GoldMine UserID, you will need to change the UserID from **(all)** to the GoldMine UserID in both History Data frame, and the Calendar Data frame.*

*And do not forget to set the **Date Range**. Typically if a report like this is going to be accessed by several users, it is recommended that you do not let the users modify the **Options** tab, and to use report dialogs instead.*

The **Options** tab is divided into different sections depending on the category. If the report is something like the **History and Pending Report** located under the **Contact Reports** category branch, then both sections need to be altered prior to running the report. Where as, if the report were strictly a History report then you would only need to adjust the options in the **History Data** frame.

The **Options** dialog form shown in Figure 12-8, on the next page, appears for all report categories with the exception of the **Calendar Printouts**, and the **Service Reports** categories, however, in **Labels and Envelopes** category, the **Options** settings do not do much for the end result. See Figure 12-5, and the detailed explanation accompanying it, for the **Calendar Printouts** category **Options**.

If the report that you want to create is very similar to an existing report, it is far easier to clone that report, and to add your customizations instead of creating the report scratch. In my experience, His-

Cloning vrs Creating New Reports

WARNING

*No where do I mention modifying a GoldMine pre-installed report. You should always **Clone** these reports as opposed to modifying them regardless of your reason.*

tory reports and Forecast Analysis reports are usually easier to create from scratch rather than cloning a report that is slightly like the end result you are after. Opportunity reports are easier to do by cloning an existing report, and altering to suit your specific needs.

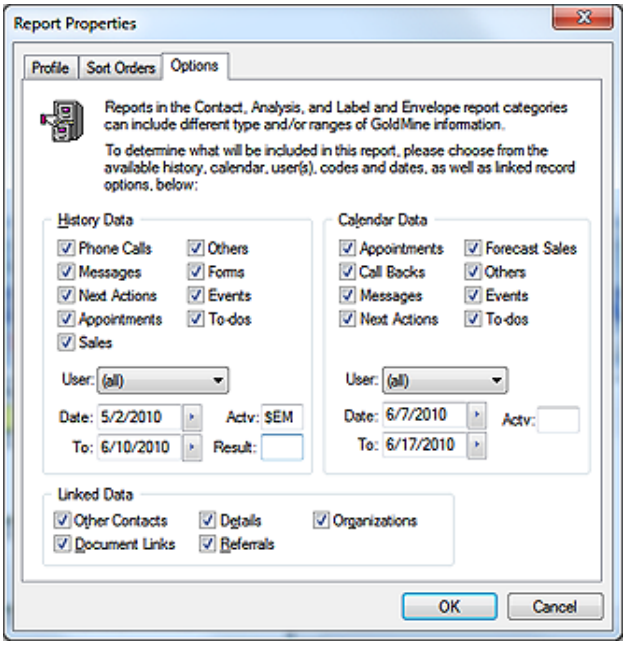


Figure 12-8

To clone a report first find the report you wish to clone, and then right-click on it. A local menu will appear, click on **Clone...**. The **Report Properties** dialog form will then open. By program default, the cloned report will appear in the title box with **Copy of** inserted in front of it, see Figure 12-9.

You may change this name at any time. The owner should be you unless you are creating this report for another specific user. Alternatively, if you listened to my suggestion earlier and created a specific Reports UserID than you may want to select that UserID from the drop list. After the report is finished, you could change it to the **(public)** user for others to use although I still don't recommend that you intermingle your custom reports within the GoldMine default reports. The cloned report will then be inserted into the same category branch from which it was cloned, but under the specified UserID. Calendar reports under **(public)** go to Calendar reports under your UserID. It is best to keep them in their categories, and to not attempt to move them to other category branches.

Note

*All that you need to do is enter a **Report Filename:** such as **Contact_List**. Click on the **OK** button, and the **Report Properties** dialog form will then create the file in the **Reports** folder under the GoldMine root folder.*

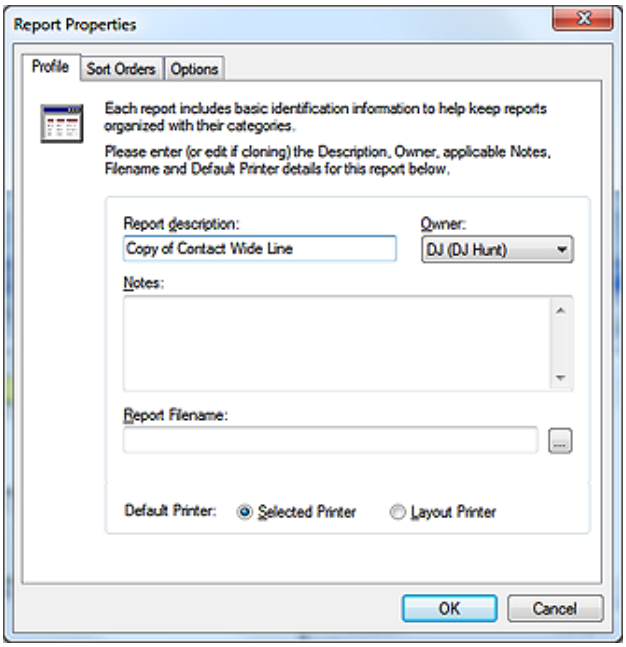


Figure 12-9

A **Report Filename:** then needs to be created. You can make it anything that you wish it to be. Pay attention to the length, it does not need to match the title name, but it should be descriptive enough so that you will recognize it when in Windows Explorer. Also, do not use any punctuation in the **Report Filename:**, the **Report description:**, however, is a whole different story.

You could put notes in the **Notes:** section if you want to, but it is not necessary. Yet it could be helpful when reviewing the report in the future. Now you need to go to **Sort Orders**, and then to **Options** to establish the first set of parameters if they are to be different than those in the original report.

To create a new report, open the **Reports Center**:

Go To
Reports ►
Reports

Select your **User:**, if it is not already present there for you, and then, under the GoldMine Reports branch, find the category branch that your report will run under.

Contact Lists, History reports belong under **Contact Reports**
Calendar Reports belong under **Calendar Reports**
Forecasted Sales, Opportunity Reports belong under **Analysis Reports**

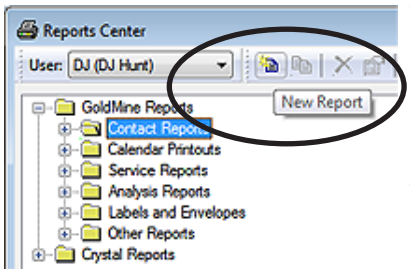


Figure 12-10

There is also the Service Reports, Labels and Envelopes, and the Other Reports branches that are available to you for this exercise. Keep in mind that what category, and what type of report you are running does matter. Different report macros are activated depending on category.

You can either right-click on the category that your new report will belong under, and select New from the local menu or click on the category to highlight it, and then push the **New Report** button in the report toolbar as shown here in Figure 12-10.

Filtering

Most people make an assumption that when one creates a report that all of the work must be accomplished by the report formulas. Truth be told, the more you can get filtering involved the faster your report will run. You can use GoldMine's own Filters and Groups for very large databases, but the report writer comes with several of its own filtering functions.

Report Filters - report filters are used when the report begins running. Dialogs may be inserted into this area as well to facilitate the correct filtering. This is how the dialog prompts appear at runtime. If you are creating a report for a particular activity type, inserting that condition into the report filters will make the report run faster.

Sort Tab - You can put a filter on the sorts, pun not intended, such as if a sort is by State, you can enter the State in both **Start at:** and the **End at:** fields. The report will return only the records that are in that range.

Tip

To create a dialog field that is more than one word, use the underscore character. To create dialog field for start date, name it start_date.

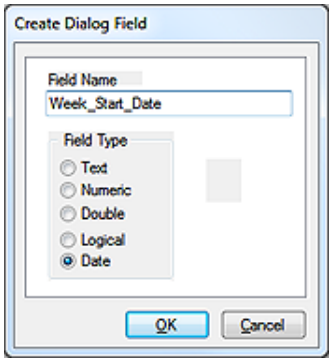


Figure 12-11

Dialog – A dialog is a prompt that is inserted into the report, and prompts the user to enter a value at runtime, such as a date range. To create a dialog, open the report layout, and place your cursor anywhere on the report. Right-click, and a local menu will appear. Select **Dialog Fields Table** ►, and then select **Create...**. The **Create Dialog Field** dialog form will appear, Figure 12-11. Create a name for the parameter, and then select the type of parameter it is.

You see that I am creating a new dialog field to be utilized in a particular report. The **Field Name** will be the actual name of the report variable so it may not contain any spaces or special characters other than the underscore as shown here in Figure 12-11.

In the next frame, **Field Type**, you have the following possible types available:

Text - string values. It is good for a UserID or the name of a city for instance.

Numeric - numeric values only. To be used for fields that are truly numeric, and not numeric text.

Double - for float values.

Logical - returns True/False, Yes/No values.

Date - for values such as CreateOn or OnDate.

Figure 12-12

To make the input dialog look nicer at runtime, you can modify the dialog field to remove the existing underscores from the actual **User Prompt:**, and replace them with spaces. Refer to Figure 12-12. To do this, right-click in the body of the report, and go to:

Dialog Fields Table ►
Modify... Ctrl+M

Now select the dialog that you wish to modify.

The name with the underscores will remain displayed in the titlebar. You can now give it a totally different name in the **User Prompt:** field which could include spaces and/or special characters.

Prompt Order: tells the report in which order this dialog input request will appear. If you are just inserting two dialogs for date range (**Start at** & **End at**) then you do not need to do anything with prompt order, the report filter will take care of it. Also the report should display prompts in the order of the report filter. There are instances when dialogs are required at different phases during runtime. If this is the case, then you should use **Prompt Order:** for all dialogs in your report.

The **Field Width:** is the standard width that is associated with the type of dialog that you had chosen. In most cases, leaving it alone is a good idea. However, you can change a date width. 8 is for MM/DD/YYYY, 6 would be for MM/DD/YY, etcetera.

Just creating the dialog is not enough. You must now tell the report to use it. Right-click on your report, and, from the local menu, select:

Report Settings ►
Filter... Ctrl+F7

Now you should see the **Report Selection Criteria** dialog form.

Report Filter Dialog examples:

Returning History records for a Date Range

Create a dialog field for **Start_Date**	Date
Create a dialog field for **End_Date**	Date
Create a dialog field for **UserID**	Text

Then insert the following into the Report Filter:

```
(ContHist->OnDate >= dlg->Start_Date .and. ContHist->OnDate <= dlg->End_Date)
```

This will return all records that where the ContHist.OnDate is within the date range as specified by the user running the report in the dialog fields.

Now let's add the dialog field for UserID into this report filter.

```
(ContHist->OnDate >= dlg->Start_Date .and. ContHist->OnDate <= dlg->End_Date) .and. ContHist->UserID = dlg->UserID
```

When the report starts to run this time, it will ask for date range first, and then the UserID for which to look. You could insert the UserID first, but you may see a performance difference at runtime unless it is a single user database.

Tip

*If you can get your users to follow the structure of the **&CalActvName** and **&ContHistActvName** macros, then you can use that instead of writing out the translation.*

Tip

If the field that you are creating your dialog on is a string field yet contains numeric text, do not use a numeric dialog unless you want to create a formula to convert the numeric text into a numeric value and then attach your dialog to that formula.

Formulas & Expressions

Returning Specified Activities

This gets a little trickier because the database itself does not know what Appointment means, and unless you can get the end user to understand to type an A then you need to convert the values at runtime in the report filter using the if/then/else functionality.

First create a text dialog for Activity. Then, in this report filter, insert the following:

```
.if. dlg->Activity = [Appointment] .then. ContHist->sRecType = [A] .else.if. dlg->Activity = [Phone] .then. ContHist->sRecType = [C] .else. ContHist->sRecType $ [D O T M]
```

Formulas and expressions can be used to return numerical data, such as the total number of activities. Expressions can be used to return boolean values, such as if a primary contact name is Joe then display on the report Joseph or True/False values. These are just two simple examples of what you can do with formulas and expressions, the list below will give a far better definition.

When to use a formula, and when to use an expression? The basic rule of thumb is; if you do not see the function listed in the builder then more than likely it will need to be an expression.

Creating Formulas

Insert ►
Expression Field F3

Open the formula builder (which is called **Calculation Field Name**, and shows up in the report as a CALC field), and the first thing that it will ask is to name your formula. You cannot have any spaces in the formula name similar to the dialog field name, but it is possible to make it look like it is more than one word by using the underscore character. So, instead of Count the Records, create the name as we have, for example, see Figure 12-13 titlebar, **COUNT_EM**.

Note

*If you are new to dbase expressions, it is a good idea to use the formula builder so you can get familiar with how the syntax is structured. It is generally not a good idea to have spaces in your formula unless you are telling it to look for a space, however, GoldMine will automatically remove any spaces that you may have added for readability. The space shown in the expression in Figure 12-13, for instance, will be removed when one clicks on the **OK** button.*

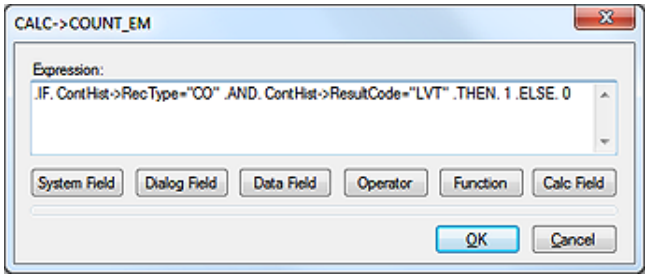


Figure 12-13

Push the **OK** button, and the formula builder opens, Figure 12-13. You could type your expression in the window or use the buttons below the expression window to assist you in your expression building.

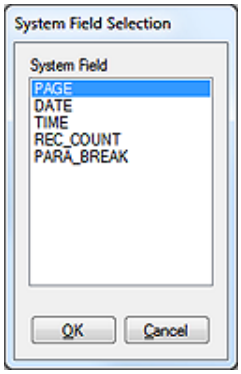


Figure 12-14

System Field - Figure 12-14, is used to insert print **Date**, **Time**, or the **Page** number into your expression. You could also insert a **Paragraph Break**, and a total **Record Count** for the criteria met on the report. Some of these items would, most likely, not be utilized in an expression as much as in the report itself.

Dialog Field - Figure 12-15 - any dialog (user entry prompt) that is created for this report will be listed in the **Select Dialog Field** dialog form for insertion into the expression that you are building for this report. You cannot create a dialog from this window, and your dialogs must have been created prior to bring up this dialog form for insertion of a dialog field into your expression.

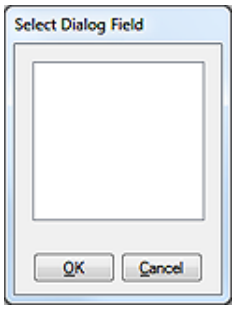


Figure 12-15

Data Field - Figure 12-16 - database fields (refer to other sections of this book for field definitions), and macros that may be utilized in your expression. There is one extra option in the **Database Name:** drop list that is not a table alias, and it is called **Macros**. Macros are like shortcuts. Such as you can use **&FullAddress** in your report instead of manually inserting each field that would create the full address. Below are the macros that can be used when inserting a new report into the **Contacts** category or in your expression, however, even though I am showing you these through the expression builder dialog form, you would probably not utilize any of the macros in your expression.

Figure 12-16

&Sort1 - displays the sort1 label

&Sort2 - displays the sort2 label

&Sort3 - displays the sort3 label

&SourceFile - the database that is being used in the report

&CalActvName - list the activity type such as Appointment from the Cal table

&HistActvName - list the activity type such as Appointment from the ContHist Table

&Contact,Company - displays Contact, and the Company name in that order, separated by a comma

&Company,Contact - displays Company, and the Contact name in that order, separated by a comma

&Name&Address - lists the Contacts name along with the Address

&Title&Address - lists the Title value along with the Address

&Company&Address - lists the Company name along with the Address

&FullAddress - list entire address including the Country field

&Address1&2 - list Address lines 1, and 2

&CityStateZip - list City, State and Zipcode with the City State values separated via a comma

&Phones - lists all 3 phone fields

&User - the GoldMine UserID for the user running the report

&FullName - the GoldMine users full name who is running the report

&Licensee - lists the name to which GoldMine is licensed

&EmailAddress - lists the primary E-mail Address for the Contact record that is currently being reported upon

When you are creating a **Pending Activity Report** that you have inserted into the **Calendar** category, the below macros could be used as well as the previously listed with the exception of sorts because it is no longer a tab option.

&Detail1 - displays information entered in the Options tab - such as UserID, Date Range, and Activities

&ActvName - lists the activity name

&Year - displays current year in a 4 digit format. For instance, if the field says 9/20/07, the report displays 2007

&Month - displays numeric value for the month. September would be displayed as 9

&CMonth - displays a text value for the month. 9/1/07 would display as September

&CMonthYr - displays the Month along with the 4 digit year. 9/1/07 would display as September 2007

&Date - displays current date, however, it could be used in date range expression

&DoW - displays the numeric day of week value for the date in question. Sunday being first day of the week would return a 0, whereas, Monday would display as a 1

&CDoW - displays the text day of week similar to DOW, 1 would display as Monday

&DayNo - works on calendar year. 9/20/2007 would display as 263

&WeekNo - Works on calendar year. 9/20/2007 would display as 38

&WDay1, **&WDay3**, **&WDay4**, **&WDay7** - returns a numeric date value. WDAY1 = Monday, 3 = Wednesday, 4=Thursday, 7=Sunday. If the record that is being reviewed by the report falls on a Tuesday, Sept. 4th, and if you are using &WDay1, the report will display a 3. By using the same example, &WDay3 would display 5, &WDay4 would display 6.

&Hour, **&Time**, **&Duration** - will all display in a 12 hour format

&FrDate, **&ToDate** - From Date, and To end Date respectively

&LastMonth, **&ThisMonth**, **&NextMonth** - all displays Cal records from last month, this month or next month respectively

Operator - contains standard operators such as equal to (=), greater than (<) as well as more complex operators. Refer here to Figure 12-17.

Figure 12-17

.total-of. - sums of numeric field. You should never place a formula with this type of operator in a header of a report because the records need to be read first. Place it in the details section to get a running total or in a Sort Footer for a group total, or the Report Footer for a grand total.

$ - The dollar sign operator tells the formula to look for a contained value. If you were looking for a value such as a city name that contains Red, then use the $ operator.

.count-of. - This counts the number of records processed. If placed in a Sort Header/Footer, it will count actual records that the sort is set to.

.if., **.then.**, **.else.** - These will probably be the most utilized operators, and they will be used together. If this happens, then do this or else do this.

The following operators work best in a Footer in the report, Sort for a total of the group, and Report Footer for a grand total:

.ave-of. - provides average of numeric fields. First it sums up the field then divides it by the number of records that met the criteria.

.max-of. - looks at all values in the met criteria, and returns the highest value

.min-of. - looks at all values in the met criteria, and returns the lowest value

Function - Function tells the formula what should be done. You should refer to Figure 12-18 on the next page.

The following items cover functions, insert the field in place of (string), (num), (date). So if you wanted to convert Company name into uppercase in your report it would look like upper(Contact1->Company)

String - text field format

Num - numeric field format (not numeric text, for numeric text, you will also have to convert it to numerical value at runtime)

Note

If you are not sure of what type of field you are trying to create in your formula, then either ask your administrator or go to user defined fields window.

***C** = string*
***N** = numeric*
***D** = Date.*

WARNING

*Do not, under any circumstance, open the properties of a field from the **User Defined Fields** dialog form. The system will **delete** all of the data that was entered.*

Note

*It is good practice to use the function **upper()** in your report to convert fields into the same case when your data input is dirty or the database indexing is not quite 100%.*

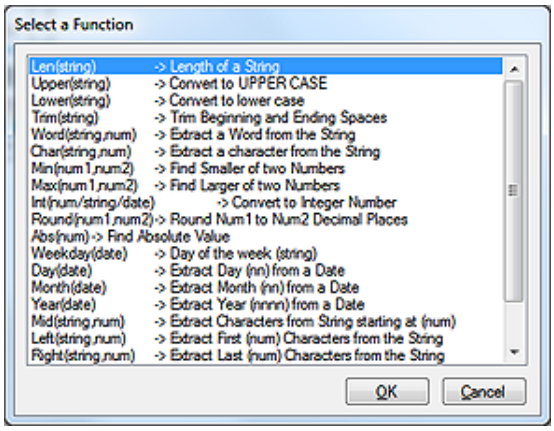


Figure 12-18

Date - date field format

Len(string) - measures length of value. Example; Andrea will be displayed as the numeric 6.

Upper(string) - converts the value into upper case.

Lower(string) - converts the value into lower case.

Trim(string) - removes extra spaces from the value.

Word(string, num) - extracts a word from a field

Char(string, num) - extracts a character from a field

Min(num1, num2) - finds the lower value of 2 numeric fields

Max(num1, num2) - finds the highest value of 2 numeric fields

Int(num/string/date) - converts field value into an integer number

Round (num1, num2) - rounds number to 2 decimal places

Abs(num) - returns the absolute value of a numeric value

Mid(string, num) - returns bytes beginning at the specified position. Example; mid("Andrea", 3) would return drea.

Left(string, num) - returns the left most bytes up to the specified number. Example; left("Andrea", 3) would return And.

Right(string, num) - returns the right most bytes up to the specified number. Example; right("Andrea", 3) would return dea.

Text(num, date) - converts the value into its string equivalent

Double(num/date/string) - converts the value into its decimal equivalent

Breaks(sum) - get the number of breaks on a sort

Date Based Functions

First thing you must do is to make sure that the machine holding GoldMine, and all workstations that will be running GoldMine are all set the same for windows long and short date formatting under the Regional Settings. That will eliminate many hours of troubleshooting when the report runs on one computer just fine, but does not run on the next computer.

WeekDay(date) - returns the day of week for the date displayed. Example; 8/1/2007 would be displayed as Wednesday on the report.

Day(date) - displays the day date for the date displayed. Example; 8/1/2007 would be displayed as 1

Month(date) - displays the month value for the date displayed. Example; 8/1/2007 will be displayed as 8.

Year(date) - displays the year value for the date displayed. Example; 8/1/2007 will be displayed as 2007

Figure 12-19

Directly Inserting Expressions

To insert a dBase expression directly into a report that a calculated field cannot handle, from the local menu:

Insert ►
Data Field F2

to bring up the dialog form shown here in Figure 12-19. Type in the expression, similar to that shown in the figure. You do not need to alter the drop down for **Database Name:** and **Field Name:** as they will be disabled anyway. The report writer will always display **Macros** and **&Sort1** respectively.

The expression, **&val(trim(Contact1->Key2))**, tells the report to remove null spaces (left and right), and then convert the

data from numeric text to a numeric value. This is useful for things such as displaying a fixed character length on your report. It is not necessary for basic math. Inserting a calc field that utilizes the **int()** function will be sufficient.

Note

A fully defined explanation/sample of each of these functions can be located in Appendix A of this book.

The following functions could be used as well, but these functions need to be inserted as an expression. You will not find them on the function menu selection:

trim() - removes spaces before and after the value entered in the field.

alltrim() - removes all spaces in the value entered.

ltrim() - removes spaces to the left of the value entered in the field.

rtrim() - removes spaces to the right of the value entered in the field.

proper() - converts the value into proper case Example; chicago is converted in Chicago (this is especially handy for reports where the data entered is not always clean data, and can be especially useful in a label or envelope reports) Caveat: proper([IBM]) would return Ibm.

date() - system date in the set date format.

substr() - returns a defined portion of the string.

ctod() - converts a character string to a date.

time() - returns the system time in hh:mm:ss format.

dtoc() - converts date field into short year format. Example; 09/12/2007 will get converted to 09/12/07.

dtos() - converts date value to string. Example; 9/12/2007 will get converted into 20070912.

reccount() - counts number of records.

stod() - converts a string value into a date value. Example; 20070912 will get converted into 9/12/2007.

val() - converts numeric text to numeric value.

str() - converts a numeric value to a string value

wdate(date, format) formats a date in different varieties.

0 - month day, 2 digit year - Sep. 12, 07
1 - day, month date, 2 digit year - Wed, Sep 12, 07
3 - long date - Wednesday, Sep 12, 2007

Note

***doy()** gets the first of the year from the date entered in the field. So for our example, 2007 is the year it looks into.*

doy() - returns the number of days that have elapsed from the beginning of the year. Example; 9/12/2007 would have displayed 255.

age() - returns the age in years. Example; a date field with 9/2/1970 will be displayed as 37.

pad() - adds extra spaces, characters to what is being displayed. Example; pad("fred", 10, "A") will display as: AAAfredAAA (fred is the value, 10 is the length you want it to be in total, A is the filler). Subsequently, you can use padl() for only adding to left of value, and padr() for adding to right of value.

Note

*Something useful about using **padl()** in a field that is good practice here. The field was 10 characters in length. The value needed to be 9 characters. The following expression was used to get the values to be 9 characters in length:*

padl(Contact2->uDunsMP,9,"0")

All that happened is the reverse, instead of adding zeros, it made what was in there disappear. There were extra spaces that were not easily visible so the expression needed to be altered:

padl(trim(Contact2->uDuns-MP),9,"0")

Always keep scenarios like these in mind when report writing. If it can happen anywhere else in the program, it can happen here as well.

iif() - iif technically means immediate if. It is typically used as a conditional expression in GoldMine, and should not be confused with the .if.. The iif() is used in expressions that are true/false. Example; iif(5 > 1, "it is bigger", "it is smaller") will return: it is bigger. .If. is used in formulas; if this happens .then. go do this. The .if. is great to use for arithmetic formulas.

Calc Field - lists the other formulas created in the report

Activities - There are two tables that hold **Activity** types, **RecType** and **sRecType**. **sRecType** is found in **ContHist** table, and it is 1 character in length, so all you need is the table shown below. For the **Cal** table, however, **sRecType** is not a choice. If you create a formula to look for Appointments in the **ContHist** table, you should not do **RecType = "A"**. It is possible for there to be more than one character in the entry. As you'll remember from the chapter on The Tables, the **ContHist.RecType** is 10 characters in length, and you need to create a formula to accommodate for the first character only unless you wanted your report to locate callbacks such as outgoing calls (**CO**) or incoming calls (**CI**). Our formula would appear as **left(ContHist->RecType, 1) = "A"**.

A - Appointment
C - Call
T - Todo
N - Next Action
O - Other Action
L - Merge form
M - Email sent

A definitive description on the **ContHist.RecType**/**Cal.RecType** and the **ContHist.sRecType** fields may be found in the chapter called The Tables found earlier in this book.

Creating Conditional Formulas

Creating a conditional formula basically means if this happens, then do this action. This is the most commonly used formula structure when reporting. You can use the formula builder selection or just type in the formula. Just remember that operators such as if, or, and, not, and else require a period before and after. Examples; **.if.** , **.or.** , **.and.** , **.not.** , **.else.**, **.then.**. These are not case sensitive, however, it does make it easier to find things in your formula down the road if you are consistent.

Formula Examples:

Count Cities - This formula says, if the city is not empty, then give it the value of 1, else give it the value of 0.

```
.if. Contact1->City <> "" .then. 1 .else. 0
```

Count Cities - This formula is counting only records where the city is equal to **Chicago**. The city field itself is much longer than the name Chicago, so by surrounding it with the trim() function tells the report to remove any extra spaces at runtime (like when the user accidentally hits the space bar before typing in the name).

```
.if. trim(Contact1->City) = "Chicago" .then. 1 .else. 0
```

Count Records - To do a count of contact records created. The reason why you count by AccountNo is because one is created for each and every record, and they are all unique. There is no chance of duplication or nulls (at least there shouldn't be). Now you can also insert an expression instead of creating this type of formula. **Reccount()** will count all Contact1 records. If you have a SQL database, the formula may give you better speed performance.

```
.if. Contact1->AccountNo <> "" .then. 1 .else. 0
```

Counting Activities - To do a count of completed **Appointments** by **User**. The formula states that if the username is not empty, and the activity type is that of an Appointment then to give it a value of 1.

```
.if. trim(ContHist->UserID) <> "" .and. ContHist->sRecType = "A" .then. 1 .else. 0
```

Text Substitution - To conditionally list a label on a report. Formulas like these are often used to display names of **Sorts** or a field that has a lookup associated with it.

```
.if. Contact1->Key4 = "" .then. "Not Assigned" .else. Contact1->Key4)
```

Find Long E-mail Address - What this formula is stating is if the **ContSupRef** is empty and the **Contact** field is equal to **E-mail Address**, then to fill using the **ContSupp.Notes** field information.

```
.if. ContSupp->ContSupRef = "" .and. ContSupp->Contact = "E-mail Address" .then. ContSupp->Notes .else. ContSupp->ContSupRef
```

Who Created the Detail Record - to find the user that entered the this particular **Detail** record.

```
left(ContSupp->City, 8)
```

What Date was the Detail Record Created - to find the date the **Detail** was entered into the **ContSupp** table, meaning when it was created.

```
left(mid(ContSupp->City, 13), 2) + "/" + left(mid(ContSupp->City, 15), 2) + "/" + left(mid(ContSupp->City, 11), 2)
```

When you create a calc expression, where you place it in your report will return different values. If your report, for example, had two sorts, **Sort1** being **State**, and **Sort2** being **City**, and your formula would be to count records. If you place the formula in the **Sort1** footer, then it will return the total number of records in the **State** being viewed. If you take that same formula and place it in the **Sort2** footer, then it will display the total number of records for the **City** being displayed.

Can you place the same formula in more than one area in your report? Yes - but it is recommended in simple reports only. If your report is going to do a lot of processing, it will run faster if you place another calc field as the grand total. If we use the example above, and the formula to count records is called **COUNT_EM** then that formula would be placed in the **Sort2** footer, and a new calc field would be made for the grand total for the **State**.

It would look like:

.total-of. calc->Count_Em

Date Month - **&month(Contact2->uDate)** - displays the numeric value for month entered in the field **uDate**.

Date Subtraction - **&ltrim(str(int((ctod("12/31/2010") - Contact2->uStartDt) / 7)))** - the first part of this expression converts the last day of the year to a numeric value, and then subtracts this value from the date entered into the field **uStartDt** and then divides it by **7**. This will return the remaining number of weeks left in the year based on the date entered.

Numeric Division - **&val(Contact2->ucPerCen)/100** – converts numeric text contained in the **ucPerCen** field into a value, and then divides it by **100** to return a percentage value.

Numeric Multiplication - **&Contact2->unMorEv * 12** - multiplies the numeric value entered into field **unMorEv**, and multiplies it by **12**.

Graphics

Most people tend to overlook making reports look pretty, and most think that the GoldMine report writer is not meant to be pretty. By taking the extra few minutes to make the report graphically appealing gives the look of professional polish, and somehow it makes it look like the report is more reliable.

Inserting Pictures or Logos - Save the image you wish to insert as a bitmap (**BMP**), and then open the report layout. If you do not have a graphics program, you can always use Microsoft Paint™ for the task (all the examples used for report writing were created in Microsoft Paint). Then push the **Picture** toolbar button when in the report layout, and the **Explorer** dialog form will open that will allow you to browse to your bitmap image.

Tip

You can shrink an image or enlarge an image only so much, as it will start losing quality. It is always good rule of thumb to save your image no more than 10% smaller than the actual size, and no more than 30% larger than the actual size.

After you have inserted the image into the report, you can also place a border around the image by right-clicking on it, and then, from the local menu, choosing:

Edit ►
Outlines... Ctrl+O

You can create a drop shadow illusion for a field by checking only the bottom from the **Outlines** dialog form, and either the right or the left side. The **Line Color** button allows you to choose the line color.

Backgrounds - When you right click on a label or field, you can select from the local menu to:

Edit ►
Background... Ctrl+B

This will allow you to change the background design/color for the label or field. Additionally, you could create a background color for each section by double-clicking on the section header, and then clicking on the **Background** button. There is no function to do the entire report, only sections and fields, hence, to do the entire report you would need to do all of the sections the same.

Tip

*Watch out for fonts that have serifs. Those are font tails like **Times New Roman**. Some video cards have a problem with the tails bleeding into each other. Also, when you choose a font, be sure it is something that all machines using this report will have installed, or you will run into problems at runtime. A report favorite is **Arial** or **Tahoma**. You also want to be careful of using narrow fonts, like **Arial Narrow**.*

Fonts - Double-click on the field or label, and selecting the **Font** button. You can change font type, size and color. **Underlining**, **Italic** & **Bold** can be set in these properties as well.

Inserting Lines and Boxes - To insert a line in your report, right-click, and select from the local menu:

Line ►
Line Ctrl+F9

Tip

You should not use more than 2 different font patterns, and you should never use more than 2 colors (white doesn't count as a color). The more fonts and colors used, the more difficult it is to read the report. You want it to look pretty, not garish.

Alternatively, and probably easier, would be to select **Line** from the **Toolbar**. There is no official function to create a box when inserting your line, but, after inserting the line, you could then double-click on the line to change its characteristics. A dialog form appears, Figure 12-20, and you can change the line property from a ◉ **Horizontal Line** to a ○ **Vertical Line**, a ○ **Diagonal Line** or even a ○ **Back Diagonal Line**. As well, from the **Line Style** frame, you could choose the style for this line. You will notice that there are also buttons that will allow you to modify

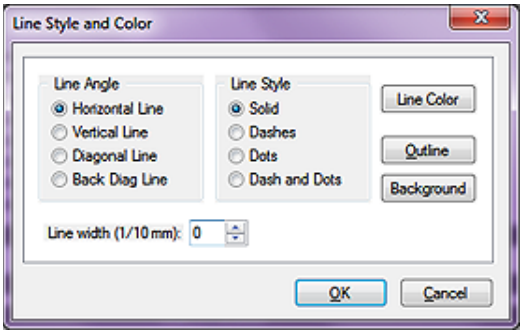


Figure 12-20

the **Line Color**, **Outline**, and the **Background** just as was mentioned previously for the fonts, and backgrounds.

You could also create an outline for any data or formula field in your report. By right clicking on the field, and, from the local menu, selecting:

Edit ►
Outlines... Ctrl+O

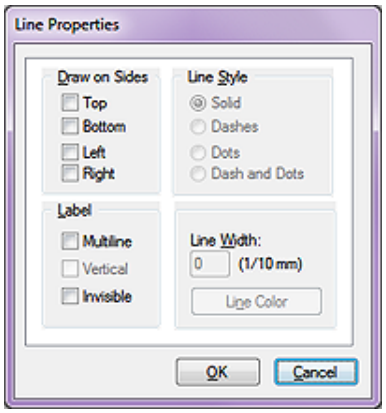


Figure 12-21

...the dialog form, shown here in Figure 12-21, will be displayed. You'll notice immediately that this dialog form has more options available for the **Line Properties** than that which you saw in Figure 12-20, previous page, on the **Line Style and Color** dialog form.

The **Draw on Sides** frame is for the **Outline** itself, and, as before, let's you select the sides upon which you wish the line displayed. Checking all four will place a box entirely around the entity.

The **Line Style** frame is a radio button selection that will define the style of the line to be used. The default will always be ◉ **Solid**.

The **Label** frame allows you to manipulate the label, and this is a checkbox selection meaning that all are selectable. Although why anyone would want a label, and then select to have it **Invisible** is beyond me. All three choices in the **Label** frame are only enabled if you right-click on a label. Each option in this frame is self descriptive.

Lastly, in the unlabeled frame, you may modify the **Line Width** in **(1/10mm)** increments. As well, you could also change the **Line Color** at this point by clicking upon the button or utilizing your hot key, **Ctrl+n**.

Report Example
Contact List Report

Overview: In this report we need to create a **Contact** listing that can be used by anyone that wishes to see the contacts in any **Category**. This report needs to print the records in the various categories by **City**. Since more than one category can be in the same city, the report will need to sort accordingly. The database has **Contact1.Key1** as the field where the category is identified in this database. We will be basing the category on the **Contact1.Key1** field value for this report.

Step 1: Open the **Reports Center**. Make certain that your name appears in the **Users:** field, if not change the value so that you are the user of record for this report, and then choose (highlight) the **Contact Reports** category.

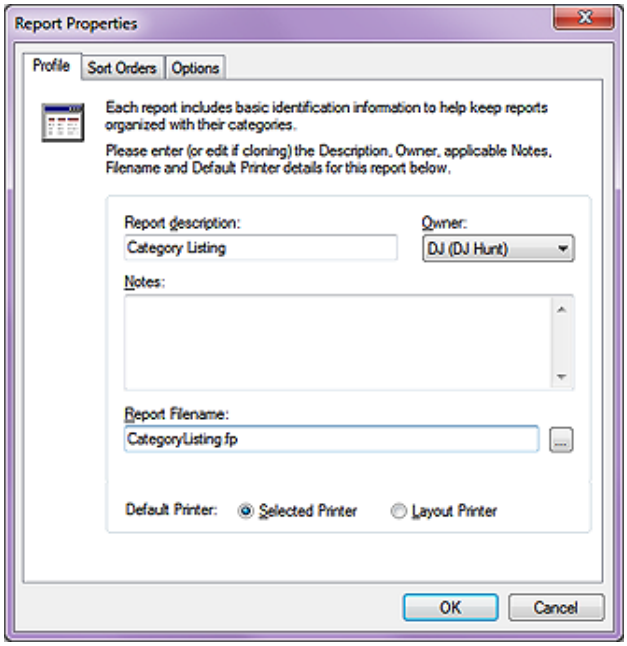


Figure 12-22

Step 2: Select the **New Report** button from the **Report Center Toolbar**, and the dialog form shown here in Figure 12-22 is brought up, and ready for your completion. Give your report a descriptive name in the **Report description:** field. Remember to make this a meaningful description that you will be able to understand its meaning if you don't revisit this report for another year. Then fill in the **Report Filename:** field. I ask you to do this so that your report file name will have some meaning as opposed to the indescript names that are auto assigned by GoldMine Premium, ~GM1008.fp, should you forget to enter anything into this field. For this report I have chosen **CategoryListing.fp**. The report file name must end in **.fp**.

You may also have noticed that I

Note

*Even though the **Sort Database:** drop list may appear to be enabled, do not be confused as it is not really enabled. You will not be able to enter a **Field Name:** unless you first pick a sort database from the drop list even if it is the same default **Contact1** table that already appears to be selected.*

have accepted the **Default Printer: ⦿ Selected Printer**. Click on the **OK** button to create this report. Next highlight the **Gategory Listing** report, and right-click on the report. From the local menu select **Options...**, and then click on the **Sort Orders** tab.

Step 3: For the **Primary Sort** select or reselect the **Contact1** table, and then, in this case, the **Category** field (Contact1.Key1). Then for the **Secondary Sort** select the **Contact1** table again, and this time select the **City** field. Click on the **OK** button, and now open the report in layout mode by clicking on the **Edit Report Layout** icon in the **Report Center Toolbar**.

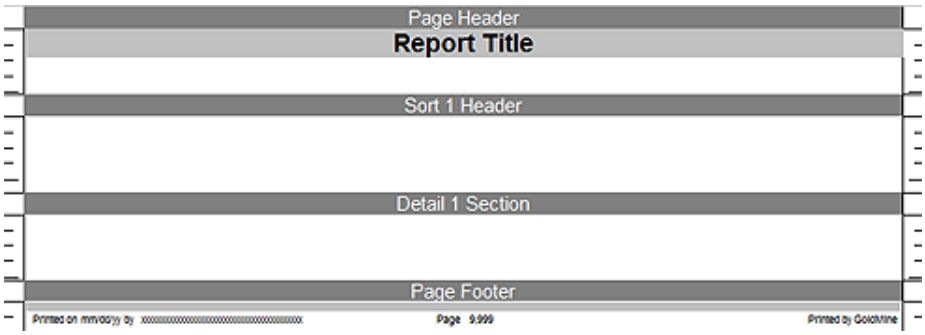


Figure 12-23

Step 4: The report should have **Page Header**, **Sort 1 Header**, **Detail 1 Section**, and a **Page Footer** already present as shown here in Figure 12-23. Let's add a section - **Sort 1 Footer**. Our list does not need to have a break in the section for each city so inserting the **Sort 2 Header** will not be necessary for this report.

Note

*Remember that we cannot have any spaces in our dialog **Field Name** when we create it, so we use the underscore key. Refer to Figure 12-24a.*

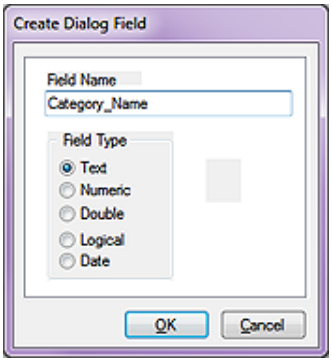


Figure 12-24a

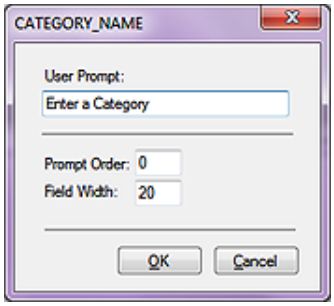


Figure 12-24b

Step 5: Now we will create a new dialog field (**Ctrl-R**), Figure 12-24a. We will give it a field name that the end user will understand, but we will also edit the field characteristics (**Ctrl-M**) to give this dialog field a more user friendly **User Prompt:**, Figure 12-24b.

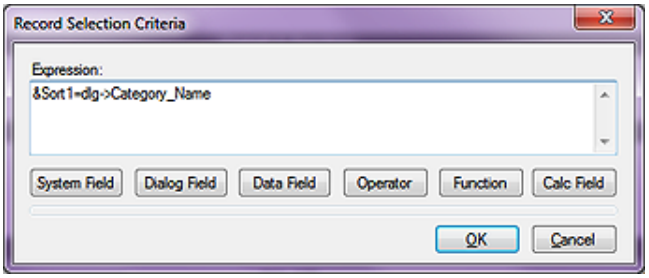


Figure 12-25

Step 6: Now let's create the general report filter (Filter button from the **Category Listing** layout Toolbar or **Ctrl-F7**). In Figure 12-25, notice the **Expression:** that I have entered. If you are following along, please enter this same expression.

Step 7: Insert the fields, and the field labels that we want to display, Figure 12-26 on the next page. You will notice that the page header has the **PAGE_HEADER** expression inserted. This particular espression that I utilized is:

```
"Category Selected: "+trim(Contact1->Key1)
```

You could also insert the **Sort1** label here as well, and you would have achieved a similar result. Since the field labels do not need to be printed for every entry, they were inserted in the **Sort 1 Head-**

er frame. If the report were multiple pages, and this needed to be on all pages, then insert these field labels into the **Page Header** frame or you could select the **Sort 1 Header** property to **Reprint titles on every page**. Remove the background, and insert a colored box for the logo background.

Step 8: We are now going to add the formula to count records to put in the **Sort 1 Footer**.

```
.if. Contact1->AccountNo > "" .then. 1 .else. 0.
```

Now insert the formula into the **Sort 1 Footer**, and right click on it. Select **Edit** | **Properties** from the local menu. This is where we can tell the field what type of summary we would like for it to do. 9 times out of 10, the **Total** summary is the proper setting, but it can never hurt to check, and, of course, the **Total** is what we desire for this particular report.

Step 9: Now to make it look pretty. First the **Page Header** was changed to a solid color background. Next the logo in bitmap format was inserted into the **Page Header**. The **Page Header** label was changed from **Category Selected: xxxxxxxx**, and the color and background was changed. Double-click on the **Detail 1 Section**, and select **Outlines**. Check the Top outline box, and change line color and thickness as is appropriate to your designing eyes. Alternatively, as DJ has done in Figure 12-26, insert the report Line. Our completed layout appear below;

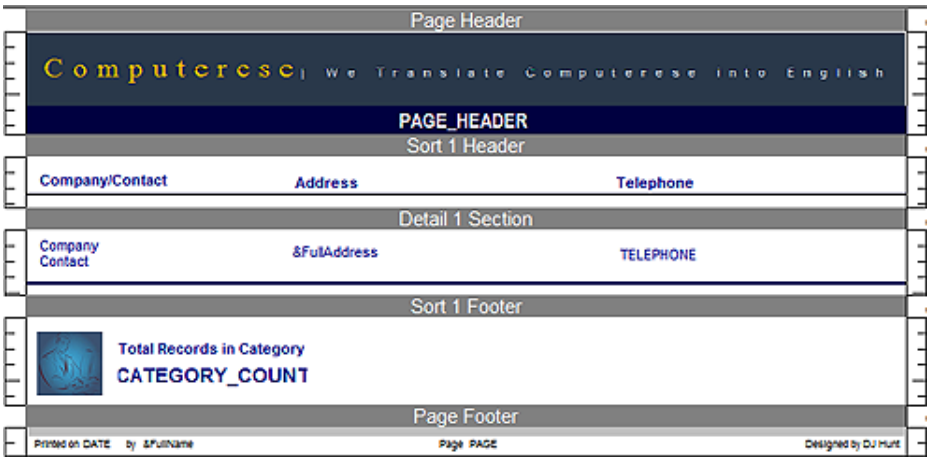


Figure 12-26

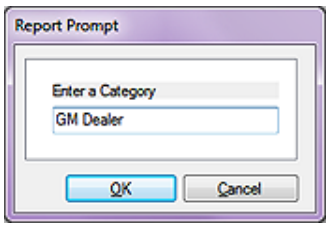


Figure 12-27

Step 10: Save the report as we are now ready to run it. Go to **Print a Report**, and change to your username (if not already selected), and double-click on your report. When the report runs, the first thing it asks for is the dialog information as shown here in Figure 12-27.

Step 11: The resulting report is shown to you in Figure 12-28 on the next page. It was way too big for this page.

Step 12: Some of the records do not have all of the phone numbers entered. This is common of most end users. You can leave them with the empty spaces or you can create a formula to point out that the phone number is missing. To do this, open the report layout, and remove the Phone1 field.

Step 13: Next create a formula:

```
.if. Contact1->Phone1 = "" .then. "Need To Fill In" .else. Contact1->Phone1
```

This tells the report if there is nothing already entered in the **Phone1** field then print on the report **Need To Fill In**.

Step 14: There are two things left to do. The first is to do a screen shot of your finished report. Save it as a JPG, and name it the same name as your report filename. Since this report file name is **CategoryListing.fp**, our screenshot file name would be **CategoryListing.jpg**. Save the jpg in the reports folder in GoldMine, and then go into the properties of your report. Now change the owner to **(public)**. It is a good idea to also make a copy of your report for backup purposes. By creating the JPG, you make a preview available to users in the customize reports menu.

Note

*Although Andrea doesn't state this, when I inserted the **&FullAddress** macro into my report, I did have to change some of the properties so that the resulting display matched hers.*

*Double-click on the **&FullAddress** macro field to bring up the properties dialog form, and make certain that **Wrap Text**, and **Variable Number of Lines** is checked.*

Note

I have to admit that my GoldMine is not configured the same as Andreas GoldMine as I use the Key1 field for Contact Type, whereas Andrea uses the Key1 for the Salespersons name. Hence, even though this dialog is asking me to Enter Sales Name, I was forced to enter Prospect to be able to show you the resulting report.

Note

*I, and this is DJ speaking, always suggest moving the GoldMine default reports to the **MASTER UserID** from the **(public) UserID**. Put any shared reports under the **(public) UserID**.*

Computerese: We Translate Computerese into English

Category Selected: Family/Friend

Company/Contact	Address	Telephone
Don Christoforo	78 Sylvan Avenue Leominster, MA 01453 USA	Need to Enter in GoldMine
Bob & Connie DeLorme	614 River Road New Bern, NC 28562-7462 USA	(252)633-1558
Aletta Chamberland	65 Sunset Avenue Plainville, CT 06064 USA	(860)747-3523
Monique Chamberland	23 Hart Place Unit 7 Plainville, CT 06062 USA	Need to Enter in GoldMine
Elizabeth Sweeney	56 S Randolph Avenue Poughkeepsie, NY 12601 USA	Need to Enter in GoldMine
Bert Sweeney	191 Sandalwood Lane Rhinebeck, NY 12572 USA	Need to Enter in GoldMine
Kelly Desgroseilliers	596 North Central Street Winchendon, MA 01475-1282 USA	Need to Enter in GoldMine
Eric Desgroseilliers	596 North Central Street Winchendon, MA 01475-1282 USA	Need to Enter in GoldMine
Corey Desgroseilliers	596 North Central Street Winchendon, MA 01475-1282 USA	Need to Enter in GoldMine

Total Records in Category
80

Printed on 03/13/10a DJ Hunt Page 6 Distributed by DJ Hunt

Figure 12-28

13

Microsoft SQL Server 2008®

In This Chapter

About SQL Server 2008

GoldMine Premium can no longer use the Firebird back end, and it is only capable of utilizing the Microsoft Structured Query Language (**SQL**) back end. A GoldMine Premium Edition license will permit the owner to configure GoldMine for the SQL Server back end. In fact, as of this writing, GoldMine Premium is delivered with Microsoft SQL Server 2008 for Workgroups with per seat Client Access Licenses (**CALs**). Most organizations, having this type of license, will and should install the SQL back end on the Server. I do recommend Microsoft SQL Server 2008 Express with Tools for your Remote installations of GoldMine.

Your organization today faces numerous data challenges. You need your people to make faster and more data-driven decisions, your developers to be more productive and flexible, and your managers to reduce their overall information technology (**IT**) budgets even as they scale your infrastructure to meet ever increasing demands.

SQL Server 2008 is designed to help enterprises address these challenges while enhancing performance in a Server 2008/Windows 7 environment. This next-generation data management and analysis solution delivers increased security, scalability, and availability to enterprise data and analytical applications, while making them easier to build, deploy, and manage.

Extending the strengths of SQL Server 2005, SQL Server 2008 provides an integrated data management and analysis solution that can help your staff do the following:

- Build, deploy, and manage enterprise applications that are more secure, scalable, and reliable.

- Maximize IT productivity by reducing the complexity of developing and supporting database applications.

- Share data across multiple platforms, applications, and devices to make it easier to connect internal and external systems.

- Control costs without sacrificing performance, availability, scalability, or security.

SQL Server 2008 advances your data infrastructure in three key areas: it makes your enterprise data more manageable, your developers more productive, and your business intelligence (**BI**) more comprehensive. It also breaks new ground in affordable pricing and licensing, upgrade paths to SQL Server 2008, and the Microsoft Windows Server System.

It is important to note that all of the above is virtually meaningless for the typical GoldMine user. In fact, most GoldMine Partners only know enough about SQL Server 2008 to get GoldMine Premium functioning, and to possibly create a SQL Maintenance Plan. Of course there is always the exception to the case, but you should be aware that, although supplied by FrontRange, that even FrontRange will not support your SQL Server 2008 needs. If you run into any issues you are expected to contact the Microsoft SQL Server Support Team for assistance.

You should follow the recommendations as put forth by GoldMine Premium, aka FrontRange Solutions. I, on the other hand, will be showing you the new connectivity setup, and use of a new GoldMine Contact Set. You see, as I have said earlier, and throughout this book, GoldMine no longer uses the Borland Database Engine bridge between the GoldMine application, and the database. New with the release of GoldMine 7 Corporate Edition was the removal of the dependency on BDE, and the application of the new bridge, **A**ctiveX **D**ata **O**bjects (**ADO**). The ADO bridge has continued through to the GoldMine Premium Edition.

Server Properties

Many people ask me how I configure SQL Server. Well, I learned by trial and error. So what I'm going to show you here is what I've learned, and do not necessarily represent the best practices. These do, however, work for me in my environment as well as for most of my clients.

Not knowing anything about SQL Server, I installed SQL Server using the default configurations. Save the fact that I installed SQL Server in **Mixed Authentication Mode**, I changed nothing. Let's take a look at the **Properties** for my SQL Server installation, and we do this from the **Microsoft SQL Server Management Studio**. I simply highlight the server name, and then right click on it to select **Properties** from the local menu which produces this dialog form:

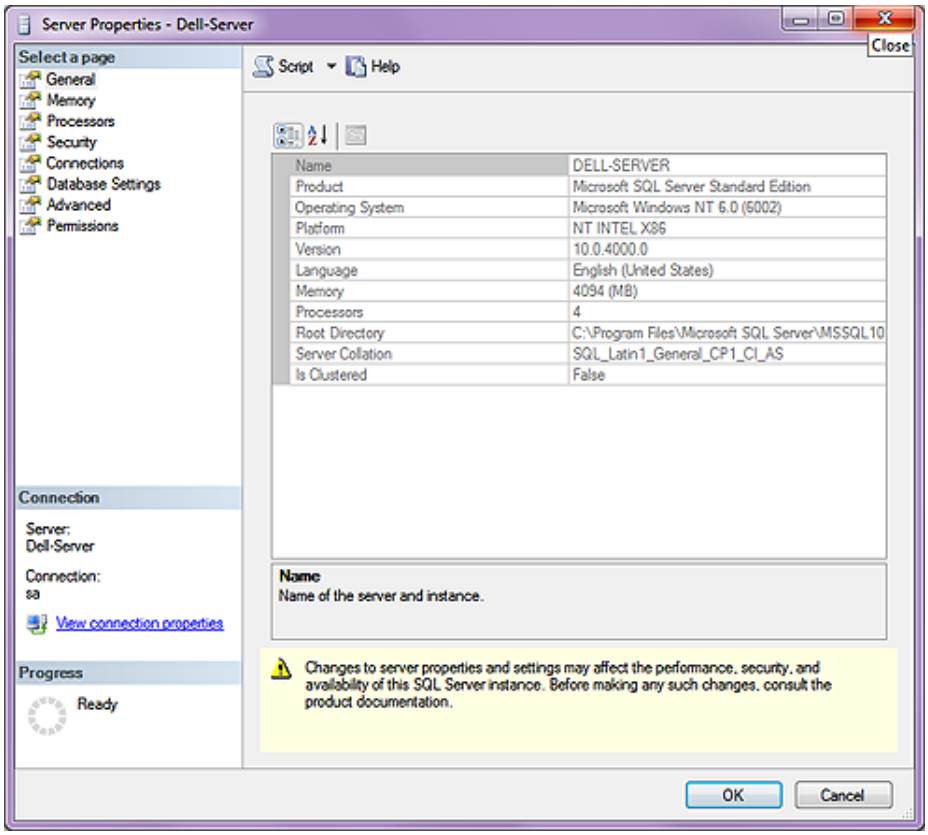


Figure 13-1

You will notice the tree to the left side of figure 13-1, and at the **General** branch is selected when you open this dialog form although, I must say, on my system, the highlighting is barely visible. You may have noticed that on the right half of the dialog form, that this information is grayed out (disabled) as this is only informational, and no changes can be made from this branch.

Once we select the **Memory** branch, however, we can see that there are items that may be modified. I won't try to explain these in any more detail than can be acquired via the Help application, however, suffice it to know that I have made no changes on this screen. Pertaining to Figure 13-2 on the next page, and taken directly from the Server Properties | Help menu item:

Server memory option

☐ **Use AWE to allocate memory**

Specifies that SQL Server will take advantage of Address Windowing Extensions (AWE) in Microsoft Windows 2000 and Windows Server 2003 to support up to 64 gigabytes (GB) of physical memory. AWE only applies to 32-bit operating systems. To use AWE you must configure Windows settings in addition to this SQL Server setting. To set this option, you must configure the lock pages in memory policy.

Minimum server memory (in MB)

Specifies that SQL Server should start with at least the minimum amount of allocated memory and not release memory below this value. Set this value based on the size and activity of your instance of SQL Server. Always set the option to a reasonable value to ensure that the operating system does not request too much memory from SQL Server and inhibit Windows performance.

Note

After all that hemming and hawing about how I have all of the default settings in my screenshots, I have to tell the truth. Since The Hacker's Guide to GoldMine Premium, I have made some tweaks to my SQL Server configuration, and this is the first.

Minimum server memory (in MB)

Default:

0 Megabyte

DJs Tweak:

1073741823 Megabyte

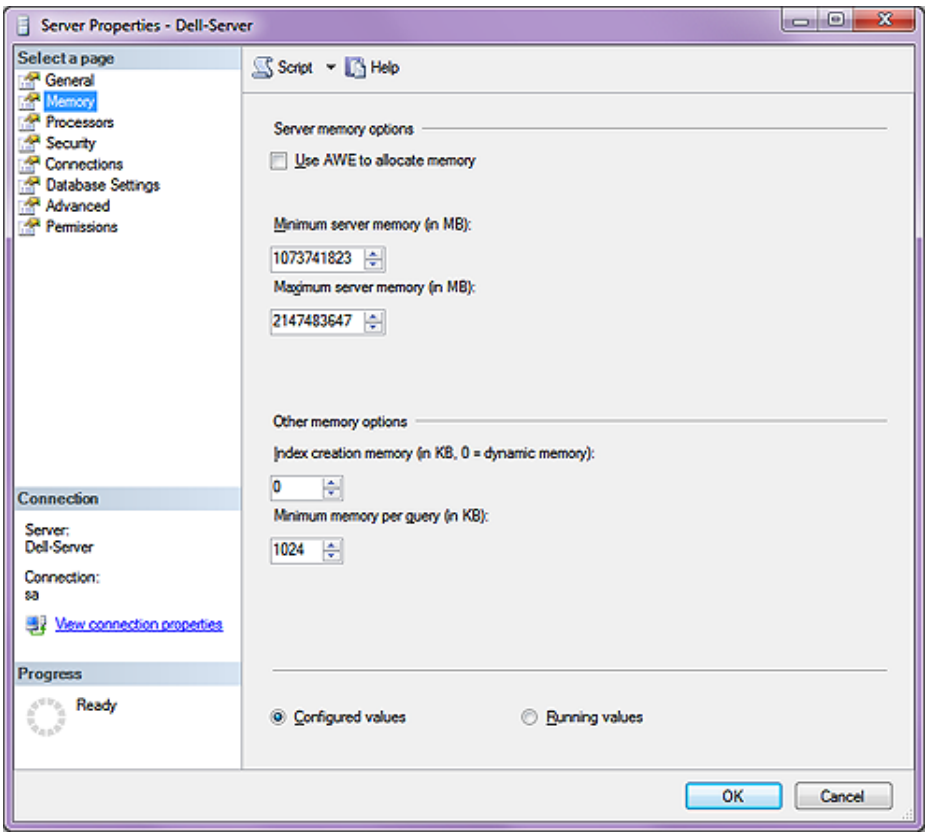


Figure 13-2

Note

Maximum server memory (in MB) *is a setting that many GoldMine Administrators have been known to tweak to their specific needs.*

Note

Index creation memory (in KB, 0 = dynamic memory)

Values from 1 to 703 are not allowed. If a value in this range is entered, the field overrides the entered value with 704.

Note

Minimum memory per query (in KB) *is a setting that many GoldMine Administrators have been known to tweak to their specific needs.*

Please note that this allocation is in Kilobytes, and not in Megabytes.

Maximum server memory (in MB)

Specifies the maximum amount of memory SQL Server can allocate when it starts and while it runs. This configuration option can be set to a specific value if you know there are multiple applications running at the same time as SQL Server and you want to guarantee that these applications have sufficient memory to run. If these other applications, such as Web or e-mail servers, request memory only as needed, then do not set the option, because SQL Server will release memory to them as needed. However, applications often use whatever memory is available when they start and do not request more if needed. If an application that behaves in this manner runs on the same computer at the same time as SQL Server, set the option to a value that guarantees that the memory required by the application is not allocated by SQL Server.

Other memory option

Index creation memory (in KB, 0 = dynamic memory)

Specifies the amount of memory (in kilobytes) to use during index creation sorts. The default value of zero enables dynamic allocation and should work in most cases without additional adjustment; however, the user can enter a different value from 704 to 2147483647.

Minimum memory per query (in KB)

Specifies the amount of memory (in kilobytes) to allocate for the execution of a query. The user can set the value from 512 to 2147483647. The default value is 1024.

◉ **Configured Values**

Displays the configured values for the options on this pane. If you change these values, click **Running Values** to see whether the changes have taken effect. If they have not, the instance of SQL Server must be restarted first.

Running Values

View the currently running values for the options on this pane. These values are read-only.

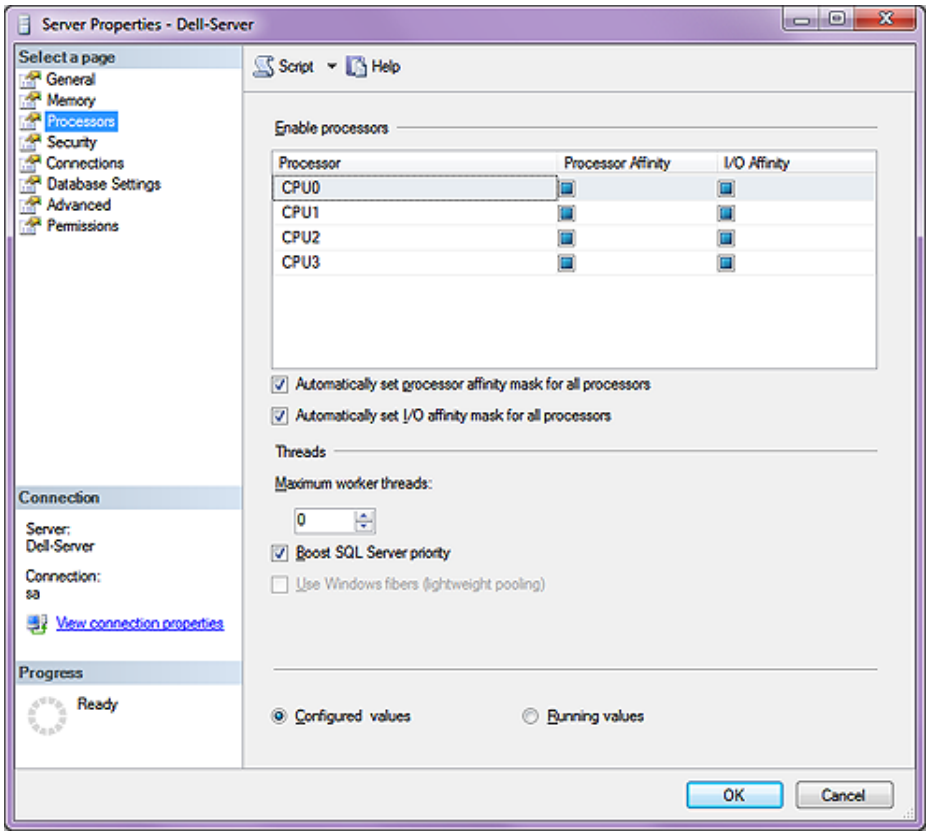


Figure 13-3

Enable processors

Processor Affinity

Assigns processors to specific threads to eliminating processor reloads and reduce thread migration across processors. For more information, see affinity mask Option.

I/O Affinity

Binds Microsoft SQL Server disk I/Os to a specified subset of CPUs. For more information, see affinity I/O mask Option.

☑ **Automatically set processor affinity mask for all processors**

Allows SQL Server to set the processor affinity.

☑ **Automatically set I/O affinity mask for all processors**

Allows SQL Server to set the I/O affinity.

Threads

Maximum worker threads

0 allows SQL Server to dynamically set the number of worker threads. This setting is best for most systems. However, depending on your system configuration, setting this option to a specific value sometimes improves performance. For more information, see max worker threads Option.

☑ **Boost SQL Server priority**

Specifies whether SQL Server should run at a higher Microsoft Windows scheduling priority than other processes on the same computer. For more information, see priority boost Option.

☐ **Use Windows fibers (lightweight pooling)**

Use Windows fibers instead of threads for the SQL Server service. Note that this is only available in Windows 2003 Server Edition. For more information, see lightweight pooling Option.

- **Configured Values**

  Displays the configured values for the options on this pane. If you change these values, click Running Values to see whether the changes have taken effect. If they have not, the instance of SQL Server must be restarted first.

- **Running Values**

  View the currently running values for the options on this pane. These values are read-only.

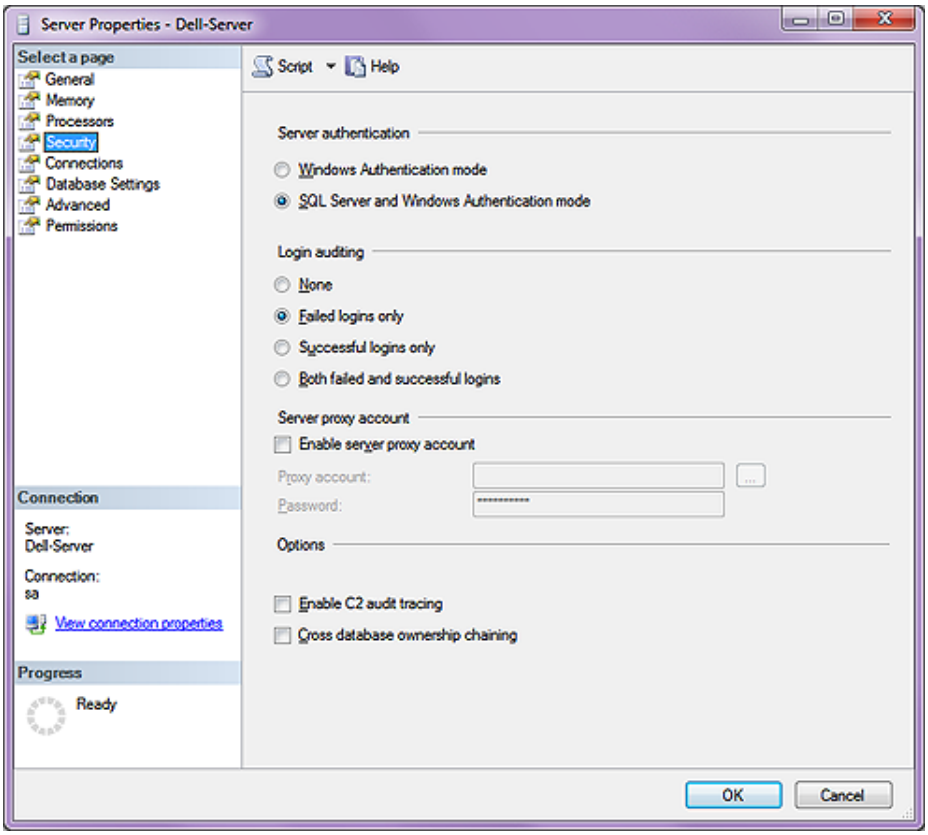


Figure 13-4

Figure 13-4 represents the **Security** branch of the tree, and its options. Most importantly, this is where I had to correct my first mistake made during my initial install of SQL Server 2005. I will again pull out the information from these Server Properties | Help application, however, I will annotate on the sidebar where I corrected my mistake.

Server Authentication

- **Windows Authentication mode**

  Uses Windows Authentication to validate attempted connections. If the sa password is blank when the security mode is being changed, the user is prompted to enter an sa password.

- **SQL Server and Windows Authentication mode**

  Uses mixed mode authentication to verify attempted connections, for backward compatibility with earlier versions of SQL Server. If the sa password is blank when the security mode is being changed, the user is prompted to enter an sa password.

Note

*Had I had chosen **Windows Authentication Mode** when installing SQL Server 2008, when in fact, I should have chosen **Mixed Mode** this, then, is where I would correct my mistake by selecting the radio button option for **SQL Server and Windows Authentication Mode**.*

*It is important to note that the **Windows Authentication Mode** is much more secure than the **SQL Server Authentication Mode**. When possible, you should use the **Windows Authentication Mode**, however, this is not possible with GoldMine Premium.*

*Changing the security configuration requires a restart of the service. When changing the **Server Authentication** to **SQL Server and Windows Authentication mode** the **sa** account is not automatically enabled. To use the **sa** account, execute **ALTER LOGIN** with the **ENABLE** option.*

Note

Changing the audit level requires re-starting the service.

Login Auditing

- ○ **None**

  Turns off login auditing.

- ◉ **Failed logins only**

  Audits unsuccessful logins only.

- ○ **Successful logins only**

  Audits successful logins only.

- ○ **Both failed and successful logins**

  Audits all login attempts.

WARNING

The login used by the server proxy account should have the least privileges required to perform the intended work. Excessive privileges for the proxy account could be used by a malicious user to compromise your system security.

Server proxy account

- ☐ **Enable server proxy account**

  Enables an account for use by **xp_cmdshell**. Proxy accounts allow for the impersonation of logins, server roles, and database roles when an operating system command is being executed.

Proxy account

Specify the proxy account used.

Password

Specify the password for the proxy account.

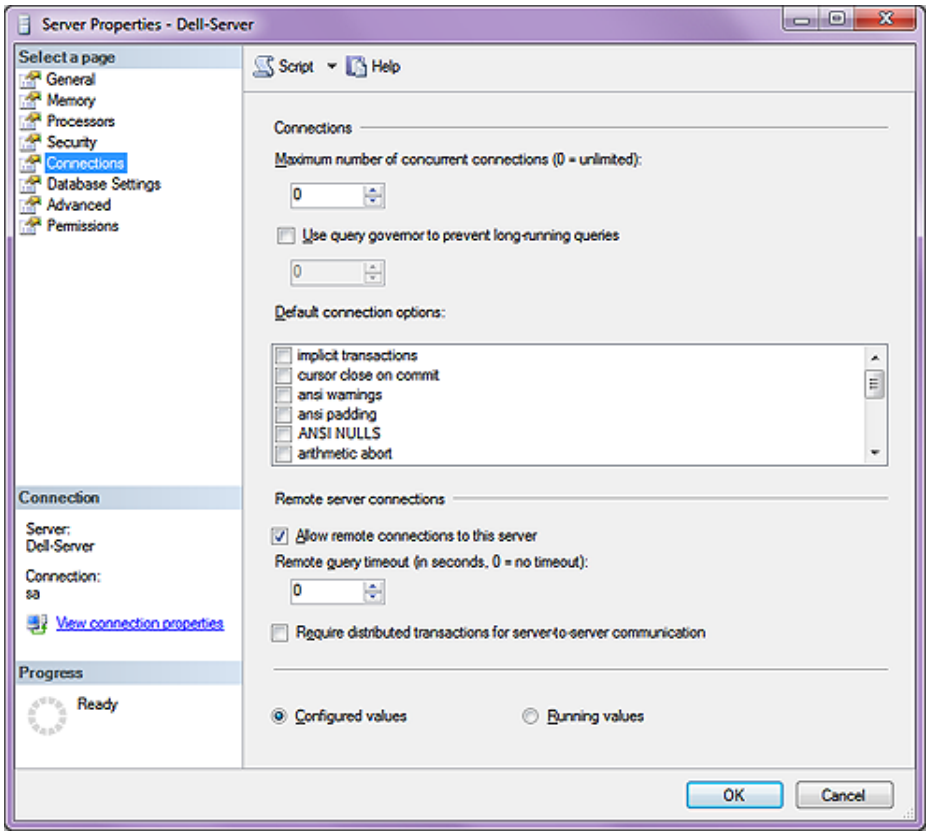


Figure 13-5

Options

☐ **Enable C2 audit tracing**

Audits all attempts to access statements and objects and records them to a file in the \MSSQL\Data directory for default instances of SQL Server, or the \MSSQL$instancename\Data directory for named instances of SQL Server. For more information, see c2 audit mode Option.

☐ **Cross database ownership chaining**

Select to allow the database to be the source or target of a cross-database ownership chain. For more information, see cross db ownership chaining Option.

Climbing on down the tree to the next branch and we come upon the **Connections** branch of the tree in Figure 13-5 on the previous page. You will notice that there are quite a few things that you can do on this dialog form. Looking again at the SQL Server | Help application for support, we find:

WARNING

*Setting this to a small value, such as **1** or **2**, can prevent administrators from connecting to administer the server; however, the Dedicated Admin Connection can always connect.*

Connections

Maximum number of concurrent connections (0 = unlimited)

If set to a value other than zero, limits the number of connections that Microsoft SQL Server will allow.

☐ **Use query governor to prevent long-running queries**

There is nothing in the SQL Help file on this option, however it appears to be self-explanatory. When selected this option defaults to **300**.

Default Connection Options

Specifies the default connection options, as described in the following table.

Configuration Option	Description
disable deferred constraint checking	Controls interim or deferred constraint checking.
implicit transactions	Controls whether a transaction is started implicitly when a statement is run.
cursor close on commit	Controls behavior of cursors after a commit operation has been performed.
ansi warnings	Controls truncation and NULL in aggregate warnings.
ansi padding	Controls padding of fixed-length variables.
ansi nulls	Controls NULL handling when using equality operators.
arithmetic abort	Terminates a query when an overflow or divide-by-zero error occurs during query execution.
arithmetic ignore	Returns NULL when an overflow or divide-by-zero error occurs during a query.
quoted identifier	Differentiates between single and double quotation marks when evaluating an expression.
no count	Turns off the message returned at the end of each statement that states how many rows were affected.
ansi null default on	Alters the session's behavior to use ANSI compatibility for nullability. New columns defined without explicit nullability are defined to allow nulls.
ansi null default off	Alters the session's behavior not to use ANSI compatibility for nullability. New columns defined without explicit nullability are defined not to allow nulls.
concat null yields null	Returns NULL when concatenating a NULL value with a string.
numeric round abort	Generates an error when a loss of precision occurs in an expression.
xact abort	Rolls back a transaction if a Transact-SQL statement raises a runtime error.

For more information on connection options, search Books Online for the specific option.

Remote Server Connections

☐ **Allow remote connections to this server**

Controls the execution of stored procedures from remote servers running instances of SQL Server. Selecting this check box has the same effect as setting the sp_configure remote access option to 1. Clearing it prevents execution of stored procedures from a remote server.

Remote query timeout (in seconds, 0 = no timeout)

Specifies how long (in seconds) a remote operation may take before SQL Server times out. The default is 600 seconds, or a 10-minute wait.

Require distributed transactions for server-to-server communication

Protects the actions of a server-to-server procedure through a Microsoft Distributed Transaction Coordinator (MS DTC) transaction. For more information, see remote proc trans Option.

Configured Values

Displays the configured values for the options on this pane. If you change these values, click Running Values to see whether the changes have taken effect. If they have not, the instance of SQL Server must be restarted first.

Running Values

View the currently running values for the options on this pane. These values are read-only.

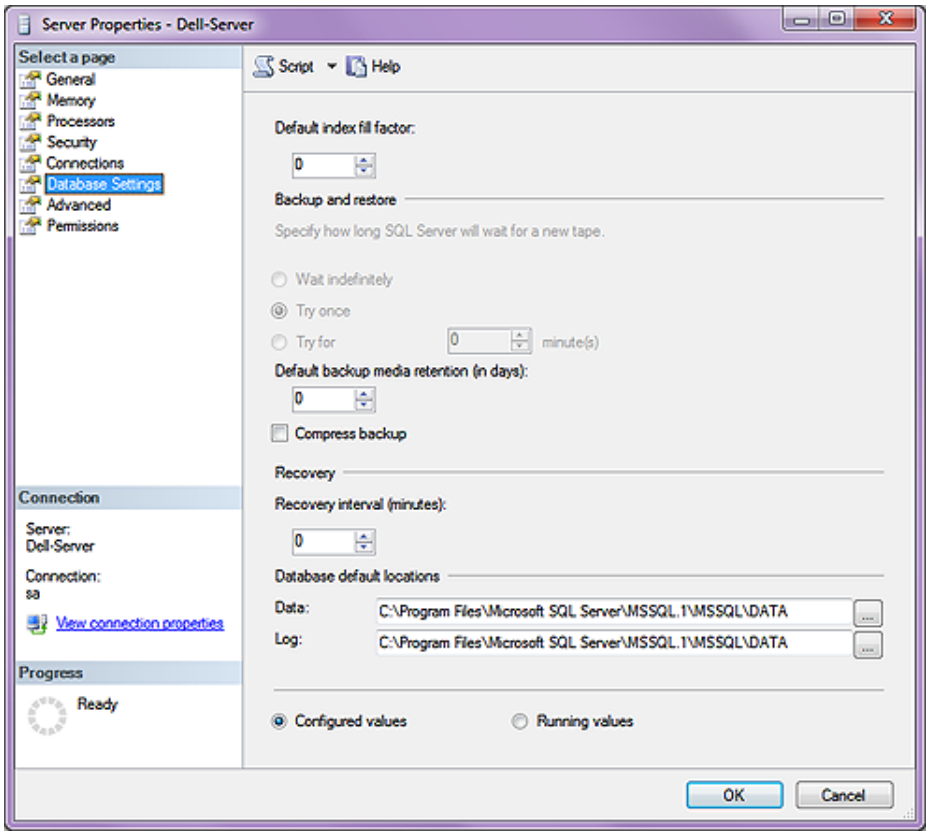


Figure 13-6

As you will see here in Figure 13-6, we have again moved down the tree to the next new branch known as **Database Settings**. Again, I will accept the comments offered by the SQL Server | Help application.

Default index fill factor

Specifies how full SQL Server should make each page when it creates a new index using existing data. The fill factor affects performance because SQL Server must take time to split pages when they fill up.

The default value is 0; valid values range from 0 through 100. A fill factor of 0 or 100 creates clustered indexes with full data pages and nonclustered indexes with full leaf pages, but it leaves some space within the upper level of the index tree. Fill factor values 0 and 100 are identical in all respects.

Small fill factor values cause SQL Server to create indexes with pages that are not full. Each index takes more storage space, but there is more room for subsequent insertions without requiring page splits.

Backup and Restore

○ **Wait indefinitely**

Specifies that SQL Server will never time out while waiting for a new backup tape.

◉ **Try once**

Specifies that SQL Server will time out if a backup tape is not available when needed.

○ **Try for minute(s)**

Specifies that SQL Server will time out if a backup tape is not available within the period specified.

Default backup media retention (in days)

Provides a system-wide default for the length of time to retain each backup medium after it has been used for a database or transaction log backup. This option helps protect backups from being overwritten until the specified number of days has elapsed.

☐ **Compress backup**

In SQL Server 2008 Enterprise (or later versions), indicates the current setting of the **backup compression default** option. This option determines the server-level default for compressing backups, as follows:

- If the ☐ **Compress backup** box is blank, new backups are uncompressed by default.
- If the ☑ **Compress backup** box is checked, new backups are compressed by default.

If you are a member of the **sysadmin** or **serveradmin** fixed server role, you can change the setting by clicking the Compress backup box

Recovery

Recovery interval (minutes)

Sets the maximum number of minutes per database to recover databases. The default is 0, indicating automatic configuration by SQL Server. In practice, this means a recovery time of less than one minute and a checkpoint approximately every one minute for active databases. For more information, see recovery interval Option.

Database default locations

Data

Specifies the default location for data files. Click the browse button to navigate to a new default location.

Log

Specifies the default location for log files. Click the browse button to navigate to a new default location.

Note

If you backup to another hard drive, as I do, you may want to consider setting this option to:

○ ***Try for x minute(s)***

WARNING

By default, compression significantly increases CPU usage, and the additional CPU consumed by the compression process might adversely affect concurrent operations. Therefore, you might want to create low-priority compressed backups in a session whose CPU usage is limited by Resource Governor. For more information, see How to: Use Resource Governor to Limit CPU Usage by Backup Compression (Transact-SQL).

Configured Values

Displays the configured values for the options on this pane. If you change these values, click Running Values to see whether the changes have taken effect. If they have not, the instance of SQL Server must be restated first.

Running Values

View the currently running values for the options on this pane. These values are read-only.

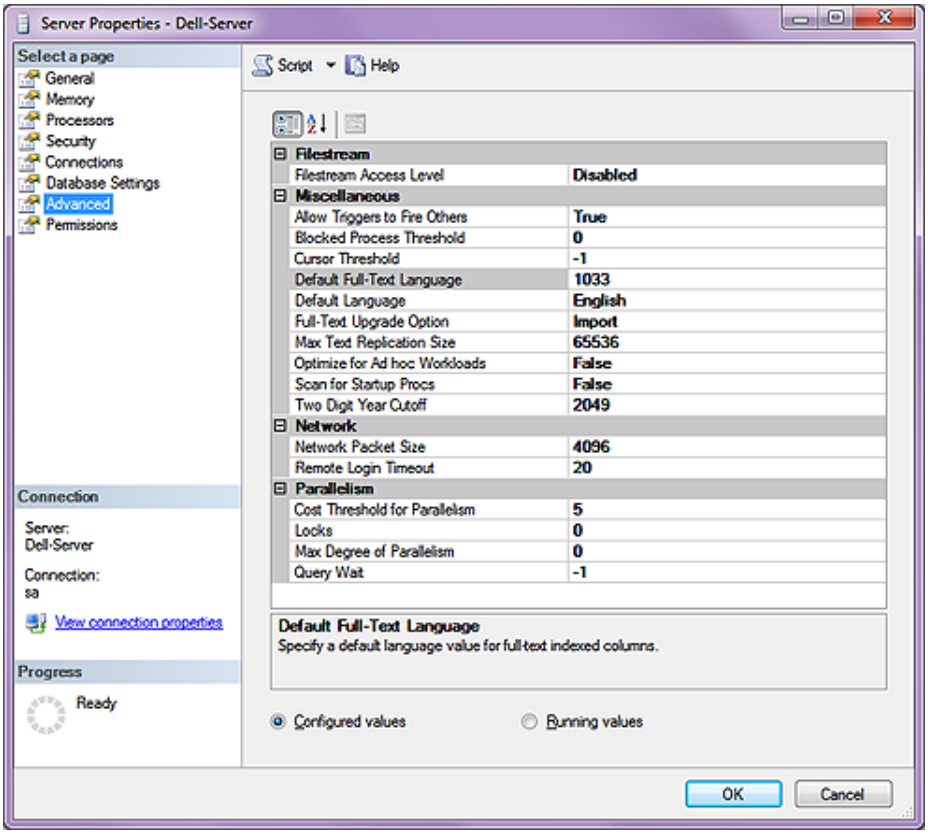


Figure 13-7

Let's climb on down to the **Advanced** branch to see options we have available to us out on this branch. Here is what the SQL Server | Help application has to say about this branch of the dialog form:

Note

When you enable FILESTREAM for the first time, you might have to restart the computer to configure drivers.

Filestream

Filestream Access Level

Shows the current level of FILESTREAM support on the instance of SQL Server. To change the access level, select one of the following values:

Disabled

Binary large object (BLOB) data cannot be stored on the file system. This is the default value.

Transact-SQL Only

FILESTREAM data is accessible by using Transact-SQL, but not through the file system.

Transact-SQL and file system (local client access only)

FILESTREAM data is accessible by using Transact-SQL and through the file system.

Miscellaneous

Allow Triggers to Fire Others

Allows triggers to fire other triggers. Triggers can be nested to a maximum of 32 levels. For more information, see the "Nested Triggers" section in CREATE TRIGGER (Transact-SQL).

Block Process Threshold

The threshold, in seconds, at which blocked process reports are generated. The threshold can be set from 0 to 86,400. By default, no blocked process reports are produced. For more information, see blocked process threshold Option.

Cursor Threshold

Specifies the number of rows in the cursor set at which cursor keysets are generated asynchronously. When cursors generate a keyset for a result set, the query optimizer estimates the number of rows that will be returned for that result set. If the query optimizer estimates that the number of returned rows is greater than this threshold, the cursor is generated asynchronously, allowing the user to fetch rows from the cursor while the cursor continues to be populated. Otherwise, the cursor is generated synchronously, and the query waits until all rows are returned.

If set to -1, all keysets are generated synchronously; this benefits small cursor sets. If set to 0, all cursor keysets are generated asynchronously. With other values, the query optimizer compares the number of expected rows in the cursor set and builds the keyset asynchronously if it exceeds the number set. For more information, see cursor threshold Option.

Default Full Text Language

Specifies a default language for full-text indexed columns. Linguistic analysis of full-text indexed data is dependent on the language of the data. The default value of this option is the language of the server. For the language that corresponds to the displayed setting, see sys.fulltext_languages (Transact-SQL).

Default Language

The default language for all new logins, unless otherwise specified.

Note

The full-text upgrade option can also be set by using the sp_fulltext_service upgrade_option action.

Full-Text Upgrade Option

Controls how full-text indexes are migrated when upgrading a database from SQL Server 2000 or SQL Server 2005 to SQL Server 2008 or later version. This property applies to upgrading by attaching a database, restoring a database backup, restoring a file backup, or copying the database by using the Copy Database Wizard.

The alternatives are as follows:

Import

Full-text catalogs are imported. This operation is significantly faster than **Rebuild**. However, an imported full-text catalog does not use the new and enhanced word breakers that are introduced in SQL Server 2008. Therefore, you might want to rebuild your full-text catalogs eventually.

If a full-text catalog is not available, the associated full-text indexes are rebuilt. This option is available for only SQL Server 2005 databases.

Rebuild

Full-text catalogs are rebuilt using the new and enhanced word breakers. Rebuilding indexes can take awhile, and a significant amount of CPU and memory might be required after the upgrade.

Reset

Full-text catalogs are reset. SQL Server 2005 full-text catalog files are removed, but the metadata for full-text catalogs and full-text indexes is retained. After being

upgraded, all full-text indexes are disabled for change tracking and crawls are not started automatically. The catalog will remain empty until you manually issue a full population, after the upgrade completes.

For information about how to choose the full-text upgrade option, see Full-Text Search Upgrade.

After you attach, restore, or copy a SQL Server 2005 or SQL Server 2000 database to SQL Server 2008, the database becomes available immediately and is then automatically upgraded. If the database has full-text indexes, the upgrade process either imports, resets, or rebuilds them, depending on the setting of the **Full-Text Upgrade Option** server property. If the upgrade option is set to **Import** or **Rebuild**, the full-text indexes will be unavailable during the upgrade. Depending on the amount of data being indexed, importing can take several hours, and rebuilding can take up to ten times longer. Note also that when the upgrade option is set to **Import**, if a full-text catalog is not available, the associated full-text indexes are rebuilt. For information about viewing or changing the setting of the **Full-Text Upgrade Option** property, see How to: View or Change Server Properties for Full-Text Search (SQL Server Management Studio).

Max Text Replication Size

Specifies the maximum size (in bytes) of text, ntext, varchar(max), nvarchar(max), xml, and image data that can be added to a replicated column or captured column in a single INSERT, UPDATE, WRITETEXT, or UPDATETEXT statement. Changing the setting takes effect immediately. For more information, see max text repl size Option.

Open Objects

Specifies the maximum number of database objects that can be open at one time on an instance of Microsoft SQL Server. Only available for SQL Server 2000.

Scan For Startup Procs

Specifies that SQL Server will scan for automatic execution of stored procedures at startup. If set to True, SQL Server scans for and runs all automatically run stored procedures defined on the server. If set to False (the default), no scan is performed. For more information, see scan for startup procs Option.

Two Digit Year Cutoff

Indicates the highest year number that can be entered as a two-digit year. The year listed and the previous 99 years can be entered as a two-digit year. All other years must be entered as a four-digit year.

For example, the default setting of 2049 indicates that a date entered as '3/14/49' will be interpreted as March 14, 2049, and a date entered as '3/14/50' will be interpreted as March 14, 1950.

Network

Network Packet Size

Sets the packet size (in bytes) used across the whole network. The default packet size is 4096 bytes. If an application does bulk-copy operations or sends or receives large amounts of **text** or **image** data, a packet size larger than the default may improve efficiency, because it results in fewer network reads and writes. If an application sends and receives small amounts of information, you can set the packet size to 512 bytes, which is sufficient for most data transfers. For more information, see network packet size Option.

Note

Do not change the packet size unless you are certain that it will improve performance. For most applications, the default packet size is best.

Remote Login Timeout

Specifies the number of seconds SQL Server waits before returning from a failed remote login attempt. This setting affects connections to OLE DB providers made for heterogeneous queries. The default value is 20 seconds. A value of 0 allows for an infinite wait. For more information, see remote login timeout Option.

Changing the setting takes effect immediately.

Parallelism

Cost Threshold for Parallelism

Specifies the threshold above which SQL Server creates and runs parallel plans for queries. The cost refers to an estimated elapsed time in seconds required to run the serial plan on a specific hardware configuration. Only set this option on symmetric multiprocessors. For more information, see cost threshold for parallelism Option.

Locks

Sets the maximum number of available locks, thereby limiting the amount of memory SQL Server uses for them. The default setting is 0, which allows SQL Server to allocate and deallocate locks dynamically based on changing system requirements.

Allowing SQL Server to use locks dynamically is the recommended configuration. For more information, see locks Option.

Max Degree of Parallelism

Limits the number of processors (up to a maximum of 64) to use in parallel plan execution. The default value of 0 uses all available processors. A value of 1 suppresses parallel plan generation. A number greater than 1 restricts the maximum number of processors used by a single query execution. If a value greater than the number of available processors is specified, the actual number of available processors is used. For more information, see max degree of parallelism Option.

Query Wait

Specifies the time in seconds (from 0 through 2147483647) that a query waits for resources before timing out. If the default value of -1 is used, the time-out is calculated as 25 times of the estimated query cost. For more information, see query wait Option.

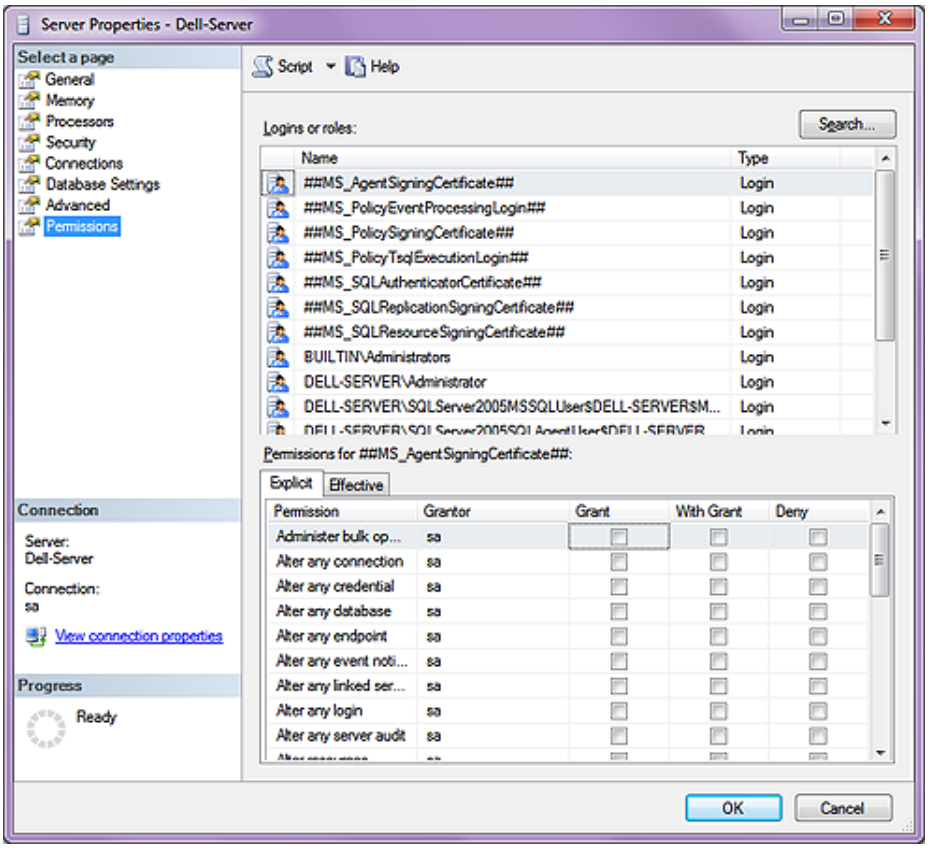


Figure 13-8

Finally we move on to the last branch of the tree, the **Permissions** branch. There is very little that I can say about this branch, but again I will give you the Help application information as displayed in SQL Server.

Use the **Permissions** page or the **Securables** page to view or set the permissions for securables. This page can be opened from many locations. The contents of the page can change slightly, depending on how the page is opened and what it contains. The top grid of the page might be populated when the page opens, or it might be empty. To add items to the upper grid, click **Search**. In the upper grid, select an item, and then set the appropriate permissions on the **Explicit** tab. To view aggregated permissions, use the **Effective** tab.

To understand the possible combinations of securables and principals, see the securable-specific syntax links in the topic GRANT (Transact-SQL). Other topics that you can refer to help you understand permissions are: Permissions Hierarchy (Database Engine), Securables, and Permissions (Database Engine).

Page Header

The header of the **Permissions** or **Securables** page varies depending on the securable or principal. It displays information relevant to the item, such as its name

Upper Grid

The upper grid contains one or more items for which permissions can be set. This dialog box provides the **Search** button for selecting objects or principals to add to the upper grid. The name of the grid might display Securables or one or more types of securables or principals. The columns that are displayed in the upper grid vary depending on the principal or securable.

Name

The name of each principal or securable that is added to the grid.

Type

Describes the type of each item.

Explicit Tab

The Explicit tab lists the possible permissions for the securable that are selected in the upper grid. To configure the permissions, select or clear the **Grant** (or **Allow**), **With Grant**, and **Deny** check boxes. All options are not available for all explicit permissions.

Permissions

The name of the permission.

Grantor

The principal that granted the permission.

Grant

Select to grant this permission to the login. Clear to revoke this permission.

With Grant

Reflects the state of the WITH GRANT option for the listed permission. This box is read-only. To apply this permission, use the GRANT statement.

Deny

Select to deny this permission to the login. Clear to revoke this permission.

Column Permissions

For objects that contain columns (such as tables, views, or table-valued functions), the **Column Permissions** button opens the **Column Permissions** dialog box. In this dialog box, you can set **Grant**, **Allow**, or **Deny** permissions on individual columns of a table or view. This option is not available for all object types or permissions.

Effective Tab

The permissions that a principal has related to a securable may come from permissions that are set for several different principals. For example, a login might be granted permissions individually and

also as a member of a group. The **Effective** tab shows the result of combining explicit permissions and the permissions that are received from group or role memberships. Grant permissions are aggregated. A deny permission overrides all grant permissions.

Permissions

The name of the permission.

Column

The names of columns that are affected by the permission.

Database Properties

And that pretty much concludes the Server Properties for SQL Server 2008. I would next like to continue on with the GoldMine SQL Database Properties. For most of the settings in here we will accept the default properties, however, there are a couple which we have chosen to tweak. Please right-click on a GoldMine database, and select **Properties** from the local menu to bring up the dialog form displayed here in Figure 13-9.

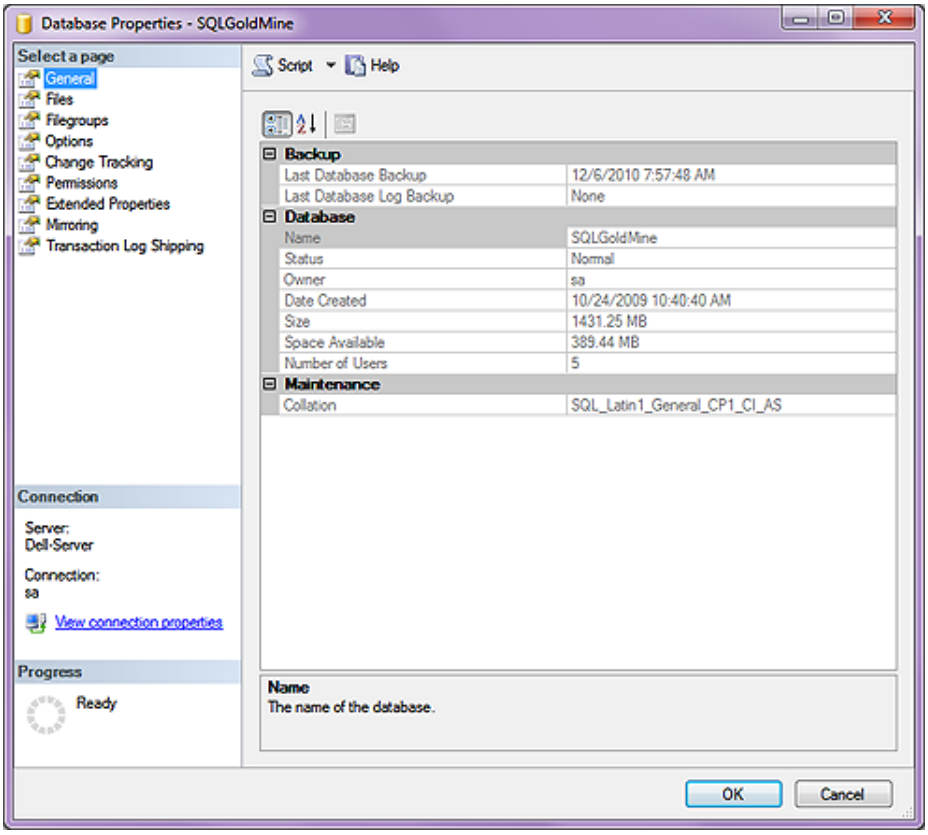


Figure 13-9

I will again work through the various tree branches in combination with the SQL Server | Help application information. Where appropriate, I will make my annotations in the sidebar. Please remember to read all of the notes contained in the sidebar as they may be very important to your SQL Server configuration. All of the information contained in the following screenshots is contained in the same live database that I have been using over the last 10 years, **SQLGoldMine**. If it works for me, it will probably work for you as well.

Beginning at the top of the tree, and working our way down the first branch that we encounter is again the **General** branch. As you can see, the dialog form is disabled (greyed out), and none of the information can be edited on this dialog page. However, here is an explanation of what these options represent:

Backup

Last Database Backup

Displays the date that the database was last backed up.

Last Database Log Backup

Displays the date that the database transaction log was last backed up.

Database

Name

Displays the name of the database.

Status

Displays the database state. For more information, see Database States.

Owner

Displays the name of the database owner. The owner can be changed on the Files page.

Date Created

Displays the date and time that the database was created.

Size

Displays the size of the database in megabytes.

Space Available

Displays the amount of available space in the database in megabytes.

Number of Users

Displays the number of users connected to the database.

Maintenance

Collation Name

Displays the collation used for the database. The collation can be changed on the Options page.

Climbing on down the tree, we next come to the **Files** branch of this tree, see Figure 13-10 on the next page. Although this dialog form does not contain that much information, it is saying much about the database.

Database name

Add or display the name of the database, and in it's default state is disabled (greyed out).

Owner

Specify the owner of the database by selecting from the list.

☑ **Use full-text indexing**

This check box is checked and disabled because full-text indexing is always enabled in SQL Server 2008. For more information, see Full-Text Search (SQL Server).

Database files

Add, view, modify, or remove database files for the associated database. Database files have the following properties:

Logical Name

Enter or modify the name of the file.

File Type

Select the file type from the list. The file type can be **Data**, **Log**, or **Filestream Data**. You cannot modify the file type of an existing file.

Note

Even though my personal database shows the **Owner:** as **sa**, many GoldMine Administrators will opt to change this to a generic like **GMUser** that has database ownership rights.

In fact, Computerese Inc has a corporate standard where we always create a **System Administrator** login of **GMUser** while assigning that user a corporate standard password as we are virtually assured that the client will forget their sa login.

Note

You'll notice that ☑ **Use full-text indexing** is selected in my configuration as we utilize the GoldMine Universal Search capabilities.

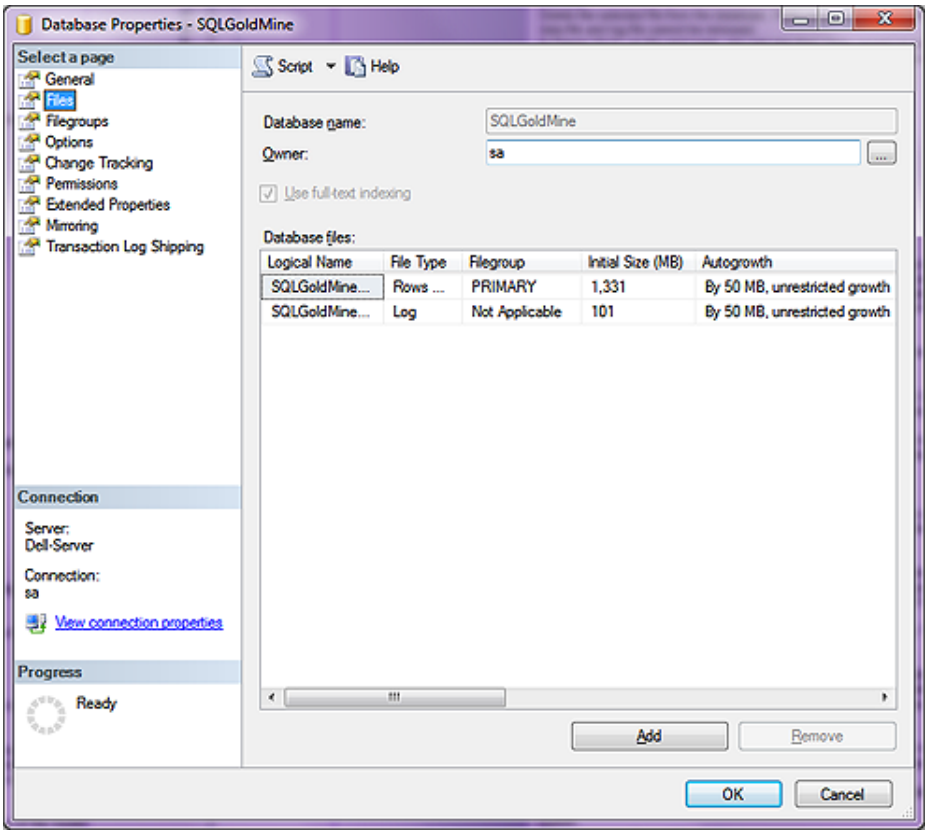


Figure 13-10

The FILESTREAM option will not appear if FILESTREAM is not enabled. You can enable FILESTREAM by using the Server Properties (Advanced Page) dialog box.

Filegroup

Select the filegroup for the file from the list. By default, the filegroup is PRIMARY. You can create a new filegroup by selecting **<new filegroup>** and entering information about the filegroup in the **New Filegroup** dialog box. A new filegroup can also be created on the **Filegroup** page. You cannot modify the filegroup of an existing file.

Initial Size

Enter or modify the initial size for the file in megabytes. By default, this is the value of the **model** database.

This field is not valid for FILESTREAM files.

Note

You'll notice that **Autogrowth** is set to **By 50 MB, unrestricted growth** on my setup.

Autogrowth

Select or display the autogrowth properties for the file. These properties control how the file expands when its maximum file size is reached. To edit autogrowth values, click the edit button next to the autogrowth properties for the file that you want, and change the values in the **Change Autogrowth** dialog box. By default, these are the values of the **model** database.

This field is not valid for FILESTREAM files.

Note

You cannot modify the path of an existing file.

Path

Displays the path of the selected file. To specify a path for a new file, click the edit button next to the path for the file, and navigate to the destination folder. You cannot modify the path of an existing file.

For FILESTREAM files, the path is a folder. The SQL Server Database Engine will create the underlying files in this folder.

File Name

Displays the name of the file.

This field is not valid for FILESTREAM files.

Add

Add a new file to the database.

Remove

Delete the selected file from the database. A file cannot be removed unless it is empty. The primary data file and log file cannot be removed.

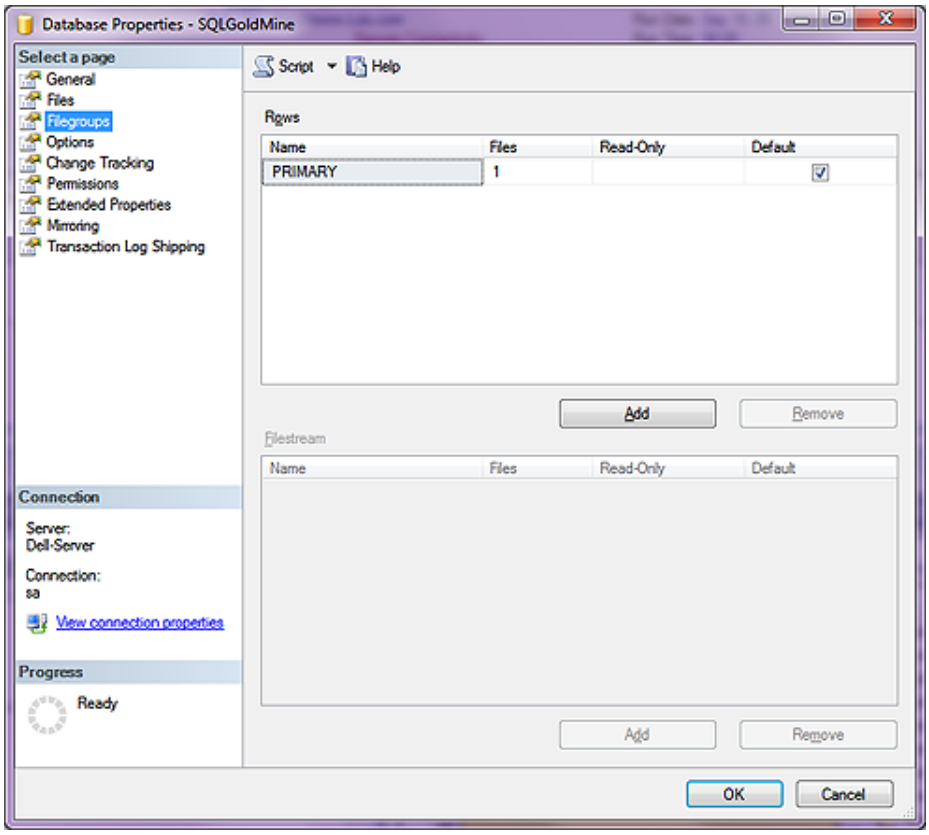


Figure 13-11

Here in Figure 13-11, you can see that I have moved on to the **Filegroups** branch of the tree, and that there is very little that can be said about this page of the dialog form. Here is what the Help application has to say about this page:

Rows

Name

Enter the name of the filegroup.

Files

Displays the count of files in the filegroup.

Read-only

Select to set the filegroup to a read-only status.

Default

Select to make this filegroup the default filegroup. You can have one default filegroup for rows and one default filegroup for FILESTREAM data.

Add

Adds a new blank row to the grid listing filegroups for the database.

Remove

Removes the selected filegroup row from the grid.

And that is all that can really be said about the **Filegroups** branch of the tree. So, let's move on to the **Options** branch of the tree, referred to here in Figure 13-12.

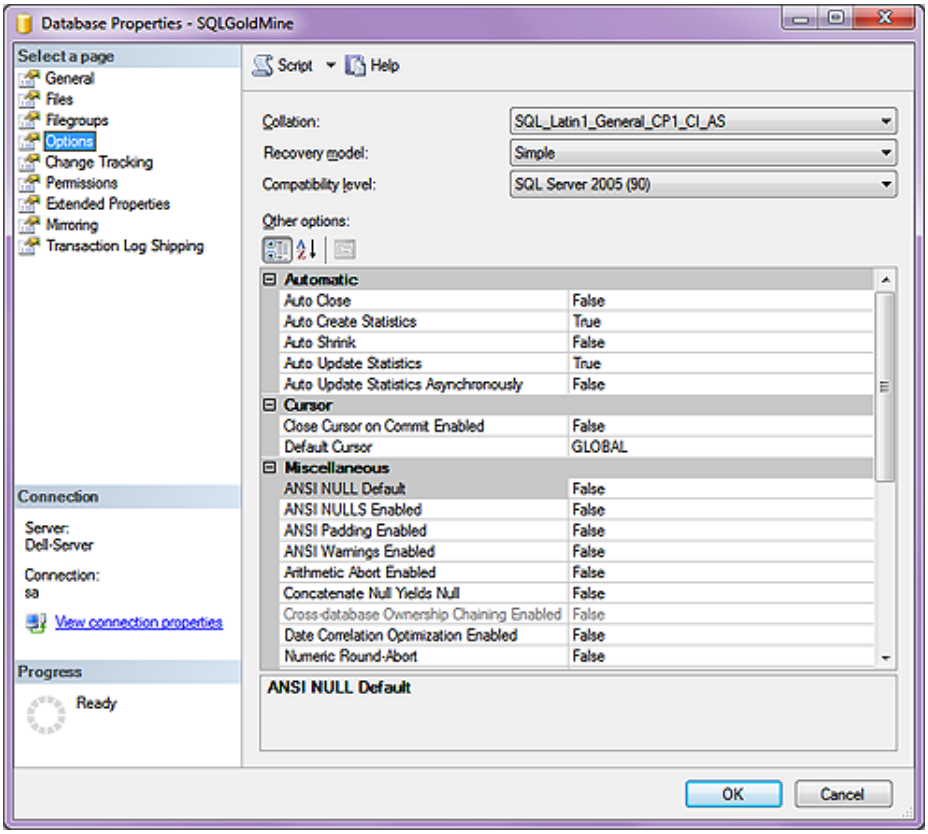


Figure 13-12

I guarantee you, that there is a lot more to be said for **Options**. Please pay attention to the sidebar notes as I have made changes to the SQL Server defaults in this area. Again, from the SQL Server | Help application we find the following information:

Collation

Specify the collation of the database by selecting from the list. For more information, see Working with Collations.

Recovery model

Specify one of the following models for recovering the database: **Full**, **Bulk-Logged**, or **Simple**. For more information about recovery models, see Recovery Model Overview.

Compatibility level

Specify the latest version of SQL Server that the database supports. Possible values are **SQL Server 2008**, **SQL Server 2005**, and **SQL Server 2000**. For more information, see ALTER DATABASE Compatibility Level (Transact-SQL).

Note

As GoldMine Premium maintains its own TLogs, we have no need for the SQL version of these logs, and we always opt for the **Simple** recovery model.

Note

If you are utilizing any 3rd party applications with GoldMine, you will probably want to leave the **Auto Close** option at **False**.

Note

You will notice that I have chosen to have the **Auto Create Statistics** to be **True**. The default would have been False, and in BDE days would have been required to have been False.

Note

Setting the **Auto Update Statistics Asynchronously** option to **True** has no effect unless **Auto Update Statistics** is also set to **True**.

Automatic

Auto Close

Specify whether the database shuts down cleanly and frees resources after the last user exits. Possible values are **True** and **False**. When **True**, the database is shut down cleanly and its resources are freed after the last user logs off.

Auto Create Statistics

Specify whether the database automatically creates missing optimization statistics. Possible values are **True** and **False**. When **True**, any missing statistics needed by a query for optimization are automatically built during optimization. For more information, see CREATE STATISTICS (Transact-SQL).

Auto Shrink

Specify whether the database files are available for periodic shrinking. Possible values are **True** and **False**. For more information, see Shrinking a Database.

Auto Update Statistics

Specify whether the database automatically updates out-of-date optimization statistics. Possible values are **True** and **False**. When **True**, any out-of-date statistics needed by a query for optimization are automatically built during optimization. For more information, see CREATE STATISTICS (Transact-SQL).

Auto Update Statistics Asynchronously

When **True**, queries that initiate an automatic update of out-of-date statistics will not wait for the statistics to be updated before compiling. Subsequent queries will use the updated statistics when they are available.

When **False**, queries that initiate an automatic update of out-of-date statistics, wait until the updated statistics can be used in the query optimization plan.

Cursor

Close Cursor on Commit Enabled

Specify whether cursors close after the transaction opening the cursor has committed. Possible values are **True** and **False**. When **True**, any cursors that are open when a transaction is committed or rolled back are closed. When **False**, such cursors remain open when a transaction is committed. When **False**, rolling back a transaction closes any cursors except those defined as INSENSITIVE or STATIC. For more information, see SET CURSOR_CLOSE_ON_COMMIT (Transact-SQL).

Default Cursor

Specify default cursor behavior. When **True**, cursor declarations default to LOCAL. When **False**, Transact-SQL cursors default to GLOBAL. For more information, see Scope of Transact-SQL Cursor Names.

Miscellaneous

ANSI NULL Default

Specify the default behavior of the Equals (=) and Not Equal To (<>) comparison operators when used with null values. Possible values are **True** (on) and **False** (off). For more information, see SET ANSI_NULL_DFLT_ON (Transact-SQL) and SET ANSI_NULL_DFLT_OFF (Transact-SQL).

ANSI NULLS Enabled

Specify the behavior of the Equals (=) and Not Equal To (<>) comparison operators when used with null values. Possible values are **True** (on) and **False** (off). When **True**, all comparisons to a null value evaluate to UNKNOWN. When **False**, comparisons of non-UNICODE values to a null value evaluate to **True** if both values are NULL. For more information, see SET ANSI_NULLS (Transact-SQL).

ANSI Padding Enabled

Specify whether ANSI padding is on or off. Permissible values are **True** (on) and **False** (off). For more information, see SET ANSI_PADDING (Transact-SQL).

ANSI Warnings Enabled

Specify ISO standard behavior for several error conditions. When **True**, a warning message is generated if null values appear in aggregate functions (such as SUM, AVG, MAX, MIN, STDEV, STDEVP, VAR, VARP, or COUNT). When **False**, no warning is issued. For more information, see SET ANSI_WARNINGS (Transact-SQL).

Arithmetic Abort Enabled

Specify whether the database option for arithmetic abort is enabled or not. Possible values are **True** and **False**. When **True**, an overflow or divide-by-zero error causes the query or batch to terminate. If the error occurs in a transaction, the transaction is rolled back. When **False**, a warning message is displayed, but the query, batch, or transaction continues as if no error occurred. For more information, see SET ARITHABORT (Transact-SQL).

Concatenate Null Yields Null

Specify the behavior when null values are concatenated. When the property value is **True**, **string** + NULL returns NULL. When **False**, the result is **string**. For more information, see SET CONCAT_NULL_YIELDS_NULL (Transact-SQL).

Cross-database Ownership Chaining Enabled

This read-only value indicates if cross-database ownership chaining has been enabled. When **True**, the database can be the source or target of a cross-database ownership chain. Use the ALTER DATABASE statement to set this property.

Date Correlation Optimization Enabled

When **True**, SQL Server maintains correlation statistics between any two tables in the database that are linked by a FOREIGN KEY constraint and have **datetime** columns.

When **False**, correlation statistics are not maintained. For more information, see Optimizing Queries That Access Correlated datetime Columns.

Numeric Round-Abort

Specify how the database handles rounding errors. Possible values are **True** and **False**. When **True**, an error is generated when loss of precision occurs in an expression. When **False**, losses of precision do not generate error messages, and the result is rounded to the precision of the column or variable storing the result. For more information, see SET NUMERIC_ROUNDABORT (Transact-SQL).

Parameterization

When **SIMPLE**, queries are parameterized based on the default behavior of the database. When **FORCED**, SQL Server parameterizes all queries in the database. For more information, see Simple Parameterization and Forced Parameterization.

Quoted Identifiers Enabled

Specify whether SQL Server keywords can be used as identifiers (an object or variable name) if enclosed in quotation marks. Possible values are **True** and **False**. For more information, see SET QUOTED_IDENTIFIER (Transact-SQL).

Recursive Triggers Enabled

Specify whether triggers can be fired by other triggers. Possible values are **True** and **False**. When set to **True**, this enables recursive firing of triggers. When set to **False**, only direct recursion is prevented. To disable indirect recursion, set the **nested triggers** server option to 0 using **sp_configure**. For more information, see Using Nested Triggers.

Trustworthy

When displaying **True**, this read-only option indicates that SQL Server allows access to resources outside the database under an impersonation context established within the database. Impersonation contexts can be established within the database using the EXECUTE AS user statement or the EXECUTE AS clause on database modules.

To have access, the owner of the database also needs to have the AUTHENTICATE SERVER permission at the server level.

This property also allows the creation and execution of unsafe and external access assemblies within the database. In addition to setting this property to **True**, the owner of the database must have the EXTERNAL ACCESS ASSEMBLY or UNSAFE ASSEMBLY permission at the server level.

By default, all user databases and all system databases (with the exception of **MSDB**) have this property set to **False**. The value cannot be changed for the **model** and **tempdb** databases.

TRUSTWORTHY is set to **False** whenever a database is attached to the server.

The recommended approach for accessing resources outside the database under an impersonation context is to use certificates and signatures as apposed to the **Trustworthy** option.

To set this property, use the ALTER DATABASE statement.

VarDecimal Storage Format Enabled

This option is read-only starting with SQL Server 2008. When **True**, this database is enabled for the vardecimal storage format. Vardecimal storage format cannot be disabled while any tables in the database are using it. In SQL Server 2008, all databases are enabled for the vardecimal storage format. For information about the vardecimal storage format, see Storing Decimal Data As Variable Length. This option uses sp_db_vardecimal_storage_format.

Recovery

Page Verify

Specify the option used to discover and report incomplete I/O transactions caused by disk I/O errors. Possible values are **None**, **TornPageDetection**, and **Checksum**. For more information, see Understanding and Managing the suspect_pages Table.

State

Database Read Only

Specify whether the database is read only. Possible values are **True** and **False**. When **True**, users can only read data in the database. Users cannot modify the data or database objects; however, the database itself can be deleted using the DROP DATABASE statement. The database cannot be in use when a new value for the **Database Read Only** option is specified. The **master** database is the exception, and only the system administrator can use **master** while the option is being set.

Restrict Access

Specify which users may access the database. Possible values are:

- **Multiple**

  The normal state for a production database, allows multiple users to access the database at once.

- **Single**

  Used for maintenance actions, only one user is allowed to access the database at once.

- **Restricted**

  Only members of the **db_owner**, **dbcreator**, or **sysadmin** roles can use the database.

Encryption Enabled

When **True**, this database is enabled for database encryption. A Database Encryption Key is required for encryption. For more information, see Understanding Transparent Data Encryption (TDE).

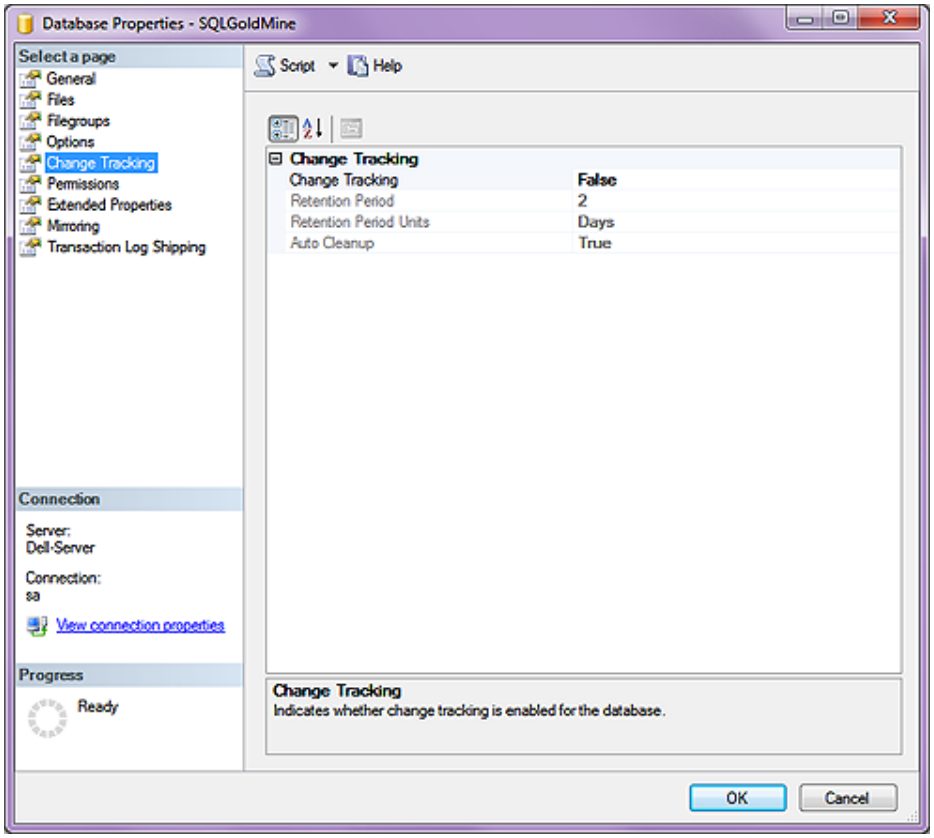


Figure 13-13

New in SQL Server 2008 is the **Change Tracking** branch with its four options.

Change Tracking

Change Tracking

Use to enable or disable change tracking for the database.

To enable change tracking, you must have permission to modify the database.

Setting the value to **True** sets a database option that allows change tracking to be enabled on individual tables.

You can also configure change tracking by using ALTER DATABASE.

Retention Period

Specifies the minimum period for keeping change track information in the database. Data is removed only if the **Auto Clean-Up** value is **True**.

The default value is 2.

Retention Period Units

Specifies the units for the Retention Period value. You can select **Days**, **Hours**, or **Minutes**. The default value is **Days**.

The minimum retention period is 1 minute. There is no maximum retention period.

Auto Cleanup

Indicates whether change tracking information is automatically removed after the specified retention period.

Enabling **Auto Clean-Up** resets any previous custom retention period to the default retention period of 2 days.

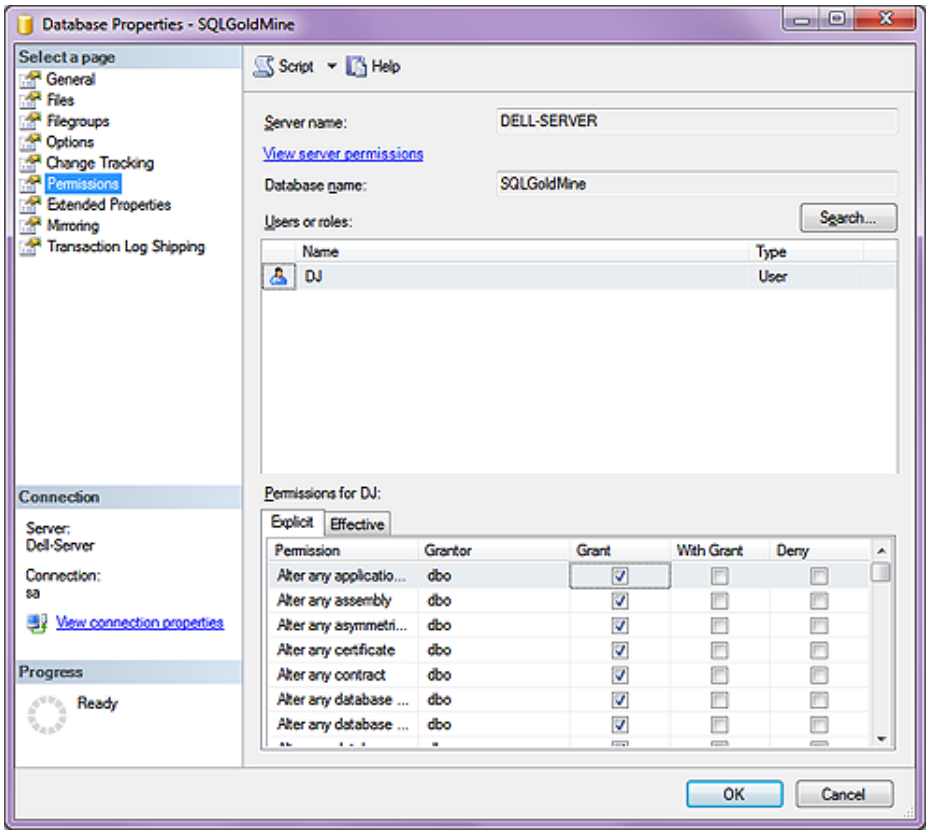


Figure 13-14

The **Permissions** branch of the tree is pretty much the same as we had discussed previously, and the screenshot of this dialog form can be seen above in Figure 13-14. There is very little that I can add to this section, so I will just take from the SQL Server | Help application:

Server name

Non modifiable field displaying the database Server name.

Database name

Non modifiable field displaying the Database name.

Users or roles

The users or roles grid contains one or more items for which permissions can be set. This dialog box provides the **Search** button for selecting objects or principals to add to the upper grid. The name of the grid might display **Securables** or one or more types of securables or principals. The columns that are displayed in the users or roles grid vary depending on the principal or securable.

Explicit Permissions

The **Explicit** tab lists the possible permissions for the securable that are selected in the upper grid. To configure the permissions, select or clear the **Grant** (or **Allow**), **With Grant**, and **Deny** check boxes. All options are not available for all explicit permissions.

Permissions

The name of the permission.

Grantor

The principal that granted the permission.

Grant

Select to grant this permission to the login. Clear to revoke this permission.

With Grant

Reflects the state of the WITH GRANT option for the listed permission. This box is read-only. To apply this permission, use the GRANT statement.

Deny

Select to deny this permission to the login. Clear to revoke this permission.

Column Permissions

For objects that contain columns (such as tables, views, or table-valued functions), the **Column Permissions** button opens the **Column Permissions** dialog box. In this dialog box, you can set **Grant**, **Allow**, or **Deny** permissions on individual columns of a table or view. This option is not available for all object types or permissions.

Effective Permissions

The permissions that a principal has related to a securable may come from permissions that are set for several different principals. For example, a login might be granted permissions individually and also as a member of a group. The **Effective** tab shows the result of combining explicit permissions and the permissions that are received from group or role memberships. Grant permissions are aggregated. A deny permission overrides all grant permissions.

Permissions

The name of the permission.

Column

The names of columns that are affected by the permission.

I won't be covering the lower three branches of the tree, **Extending Properties**, **Mirroring** & **Transaction Log Shipping** as these are pretty much read-only, however, you are welcome to delve into these on your own using the SQL Server | Help application to guide you.

SQL Server Maintenance Plan for GoldMine

Over the years people have asked me for documentation on an accepted SQL Server Maintenance Plan. To date, I have been unable to find one that was written for the GoldMine application/SQL Database. FrontRange utilizes SQL Server 2008 for Workgroups in that it distributes this with GoldMine Premium v9.0.0.110, yet to date, FrontRange will not support the SQL Server application/installation/configuration.

Computerese Incorporated has implemented many SQL Server Maintenance Plans over the years, and although we know how we do it, we really don't know if this is the correct ***FrontRange Approved*** way of doing this. I am writing this as a way of documenting the Computerese Incorporated SQL Server Maintenance Plan implementation process. As always, if you utilize these steps to implement your own SQL Server Maintenance Plan for your GoldMine Database(s), you do so at you own risk knowing full well that these steps have not received the stamp of approval from FrontRange nor any organization for that matter. Although, since it's original printing in **The GoldMine Advisor** - **August** issue, I have heard from many GoldMine Partners who have approved this process with a few exceptions which I will discuss at the appropriate time.

Step 01

Open SQL Server Management Studio, and **Connect** to your SQL Server.

Step 02

Expand the tree **+Management** by clicking upon the **+** sign.

Step 03

Right-click on the **+Maintenance Plans** and select **Maintenance Plan Wizard** from the local menu. Even though we know the steps involved, we always use the Wizard to assure a consistent Maintenance Plan execution.

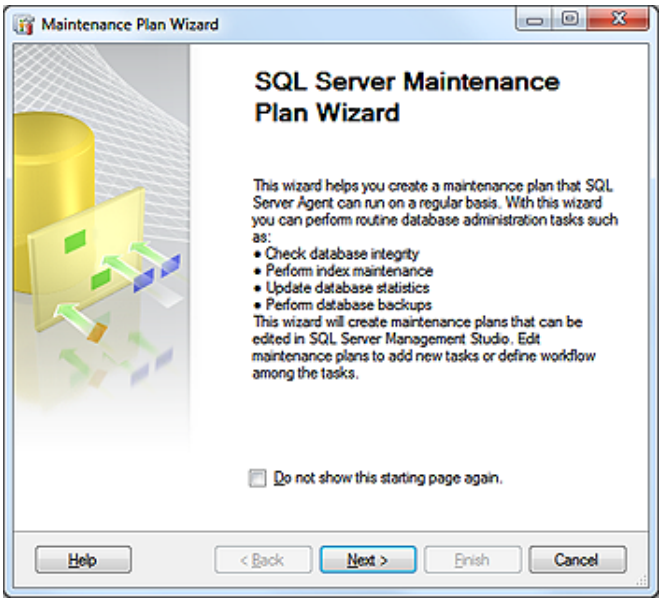


Figure 13-15

Doing so will produce the dialog form shown here in Figure 13-15. I have chosen to not check the ☐ **Do not show this starting page again.** option as I wanted you to see the default screens, however, you could select this option as this splash screen is really superfluous.

Step 04

Clicking on the **Next >** button will bring you to the **Select Plan Properties** dialog form shown here.

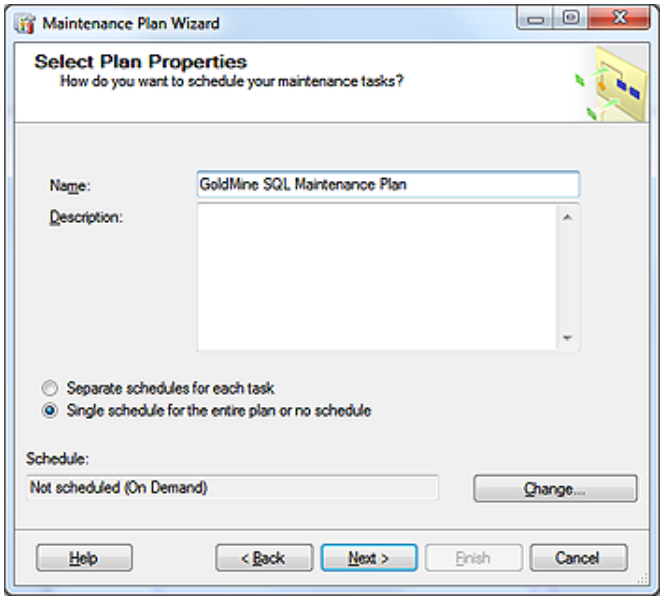


Figure 13-16

It is here that one should supply a unique **Name** for their plan. For this example I have chosen **GoldMine SQL Maintenance Plan.** Now is also the time to decide on the frequency of execution of this plan once created. One does this by clicking in the **Schedule:** area of the dialog form on the **Change** button to produce this dialog form to bring up the dialog form shown on the next page in Figure 13-17.

Now, again, this is the default screenshot, however, you must decide how to configure this based on your office backup procedures. The SQL Maintenance Plan, which can include its own Backup Plan, should not be confused with your Office Backup Plan. Redundant, yes, but how important is your GoldMine Database to your office? So important, I would imagine, that one should welcome redundancy.

Regardless, the Computerese Inc **Frequency** is **Daily**, in fact, we set up two of these plans with the first running at 12:15 pm Eastern Time, and the second running at 7:00 pm Eastern Time. We don't mind redundancy at all, and we don't plan on losing any more than 7 hours worth of work. We have clients that have requested a plan run hourly. The decision should be strictly between you and your IT department. If you plan on just nightly, make sure it executes before your Office Backup Plan so the files that you will be creating will get backed up there as well.

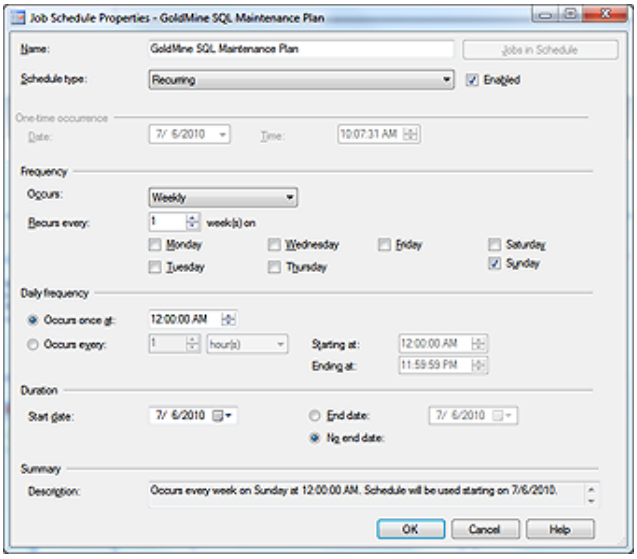


Figure 13-17

Everything else on this dialog form remains in its default state, and we would now click upon the **OK** button.

Step 05

Clicking on the **Next >** button will bring you to the **Select Maintenance Task** dialog form:

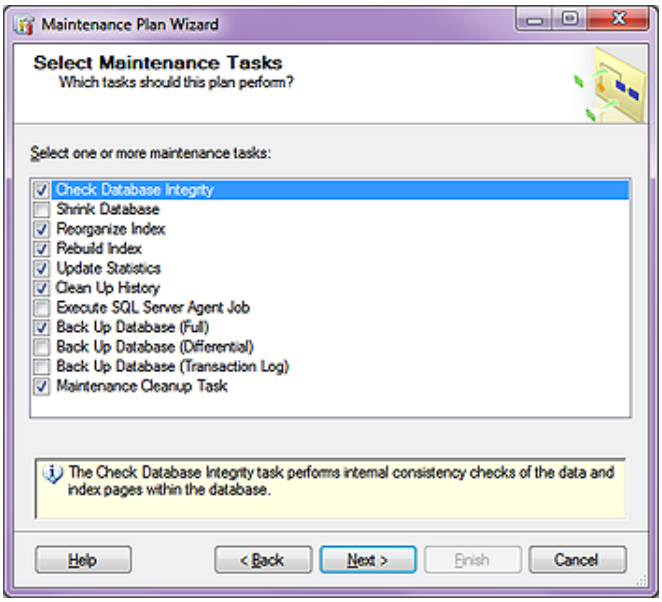


Figure 13-18

True, this is not the default state of this form, however, I did want to show you the selections that we make on this screen. I won't try to explain what each of these options represents, however, if that is of interest to you then you will find the SQL Server Help files to be most useful in this matter.

In the original article, and in the past for that matter, I had selected to ☐ **Shrink Database** as and option. Since then Jane Oliverio sent me a link that discusses the pros and the cons, actually more cons than pros, for doing this, and I have decided against shrinking my database. Here is the article reprinted from Paul S Randals' Blog in case this site goes dormant:

Note

*Readers of the original article in The GoldMine Advisor may remember that the **Shrink Database** option was selected in that article. Since that printing I have received many e-mails pointing me to sites which described why the **Shrink Database** was a bad idea. Once such article was written by one of the developers of the SQL Server application, hence, I am no longer recommending it be utilized. Said article is reprinted in this book on the next page for your review.*

Why you should not shrink your data files

by

Paul B. Randal

One of my biggest hot-buttons is around shrinking data files. Although I used to own the shrink code while I was at Microsoft, I never had a chance to rewrite it so that data file shrink is a more palatable operation. I really don't like shrink.

Now, don't confuse shrinking the transaction log with shrinking data files. Shrinking the log is necessary if your log has grown out of control, or as part of a process to remove excessive VLF fragmentation. However, shrinking the log should be a rare operation and should not be part of any regular maintenance you perform.

Shrinking of data files should be performed even more rarely, if at all. Here's why - data file shrink causes *massive* index fragmentation. Let me demonstrate with a simple script you can run. The script below will create a data file, create a 10MB 'filler' table at the start of the data file, create a 10MB 'production' clustered index, drop the 'filler' table and then run a shrink to reclaim the space.

```
USE MASTER;
GO

IF DATABASEPROPERTYEX ('DBMaint2008', 'Version') > 0
 DROP DATABASE DBMaint2008;

CREATE DATABASE DBMaint2008;
GO
USE DBMaint2008;
GO

SET NOCOUNT ON;
GO

-- Create the 10MB filler table at the 'front' of the data file
CREATE TABLE FillerTable (c1 INT IDENTITY,  c2 CHAR (8000) DEFAULT 'filler');
GO

-- Fill up the filler table
INSERT INTO FillerTable DEFAULT VALUES;
GO 1280

-- Create the production table, which will be 'after' the filler table in the data file
CREATE TABLE ProdTable (c1 INT IDENTITY,  c2 CHAR (8000) DEFAULT 'production');
CREATE CLUSTERED INDEX prod_cl ON ProdTable (c1);
GO

INSERT INTO ProdTable DEFAULT VALUES;
GO 1280

-- check the fragmentation of the production table
SELECT [avg_fragmentation_in_percent] FROM sys.dm_db_index_physical_stats (
   DB_ID ('DBMaint2008'), OBJECT_ID ('ProdTable'), 1, NULL, 'LIMITED');
GO

-- drop the filler table, creating 10MB of free space at the 'front' of the data file
DROP TABLE FillerTable;
GO

-- shrink the database
DBCC SHRINKDATABASE (DBMaint2008);
GO

-- check the index fragmentation again
SELECT [avg_fragmentation_in_percent] FROM sys.dm_db_index_physical_stats (
   DB_ID ('DBMaint2008'), OBJECT_ID ('ProdTable'), 1, NULL, 'LIMITED');
GO

avg_fragmentation_in_percent
----------------------------
0.390625
```

```
DbId   FileId      CurrentSize MinimumSize UsedPages   EstimatedPages
------ ----------- ----------- ----------- ----------- --------------
6      1           1456        152         1448        1440
6      2           63          63          56          56

DBCC execution completed. If DBCC printed error messages, contact your system administrator.

avg_fragmentation_in_percent
-----------------------------
99.296875
```

Look at the output from the script! The logical fragmentation of the clustered index before the shrink is a near-perfect 0.4%. After the shrink, it's almost 100%. The shrink operation *completely* fragmented the index, removing any chance of efficient range scans on it by ensuring the all range-scan readahead I/Os will be single-page I/Os.

Why does this happen? A data file shrink operation works on a single file at a time, and uses the GAM bitmaps to find the highest page allocated in the file. It then moves it as far towards the front of the file as it can, and so on, and so on. In the case above, it completely reversed the order of the clustered index, taking it from perfectly defragmented to perfectly fragmented.

The same code is used for DBCC SHRINKFILE, DBCC SHRINKDATABASE, and auto-shrink - they're equally as bad. As well as introducing index fragmentation, data file shrink also generates a lot of I/O, uses a lot of CPU, and generates *loads* of transaction log - as everything it does is fully logged.

Data file shrink should never be part of regular maintenance, and you should NEVER, NEVER have auto-shrink enabled. I tried to have it removed from the product for SQL 2005 and SQL 2008 when I was in a position to do so - the only reason it's still there is for backwards compatibility. Don't fall into the trap of having a maintenance plan that rebuilds all indexes and then tries to reclaim the space required to rebuild the indexes by running a shrink - that's a zero-sum game where all you do is generate a log of transaction log for no actual gain in performance.

So what if you *do* need to run a shrink? For instance, if you've deleted a large proportion of a very large database and the database isn't likely to grow, or you need to empty a file before removing it?

The method I like to recommend is as follows:

- Create a new filegroup
- Move all affected tables and indexes into the new filegroup using the CREATE INDEX ... WITH (DROP_EXISTING) ON <filegroup> syntax, to move the tables and remove fragmentation from them at the same time
- Drop the old filegroup that you were going to shrink anyway (or shrink it way down if its the primary filegroup)

Basically you need to provision some more space before you can shrink the old files, but it's a much cleaner mechanism.

If you absolutely have no choice and have to run a data file shrink operation, be aware that you're going to cause index fragmentation and you should take steps to remove it afterwards if it's going to cause performance problems. The only way to remove index fragmentation without causing data file growth again is to use DBCC INDEXDEFRAG or ALTER INDEX ... REORGANIZE. These commands only require a single 8KB page of extra space, instead of needing to build a whole new index in the case of an index rebuild operation.

Bottom line - try to avoid running data file shrink at all costs!

Note

Normally, I would not touch the arrangement on this dialog form, however, Alberto Diaz of 180° Solutions has told me that he prefers to move the **Maintenance Cleanup Task** to run prior to the **Back Up Database (Full)** task. He found, in his tests, that the plan tends to fail less frequently when executed in that order. I must say that I have had a few instances where having the **Maintenance Cleanup Task** as the last task in the sequence order has caused a failure of the Maintenance Plan to produce a **Successful** log entry.

Your mileage may vary!

Step 06

Clicking on the **Next >** button will bring you to the **Select Maintenance Task Order** dialog form.

This dialog form permits you to change the order of execution of the steps which you had selected on the previous dialog form. Review the Notes before proceeding to Step 07.

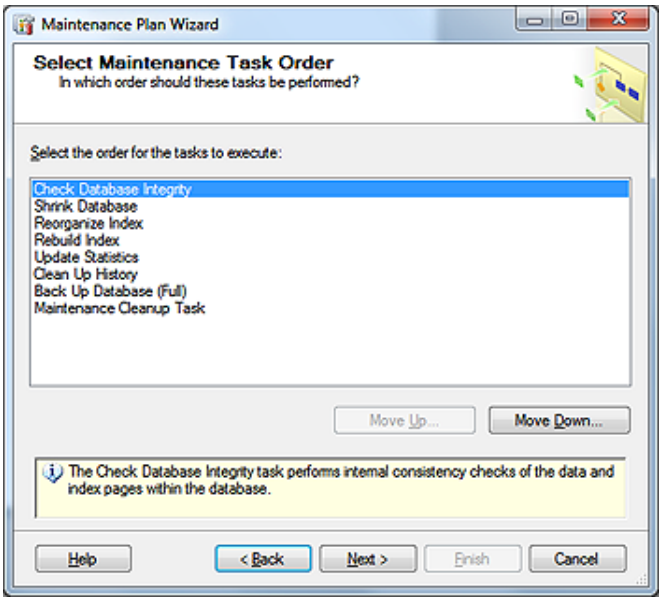


Figure 13-19

Step 07

Clicking on the **Next >** button will bring you to the default screen of the **Define Database Check Integrity Task** dialog form shown here in Figure 13-20.

However, in the default state, the **Database:** input field would have stated: **<Select one or more>**, and it would have been up to you to select one or more databases to include in this plan. As this is the same dialog form that is displayed on each of the successive dialog forms, I will show it to you but only this once in this article.

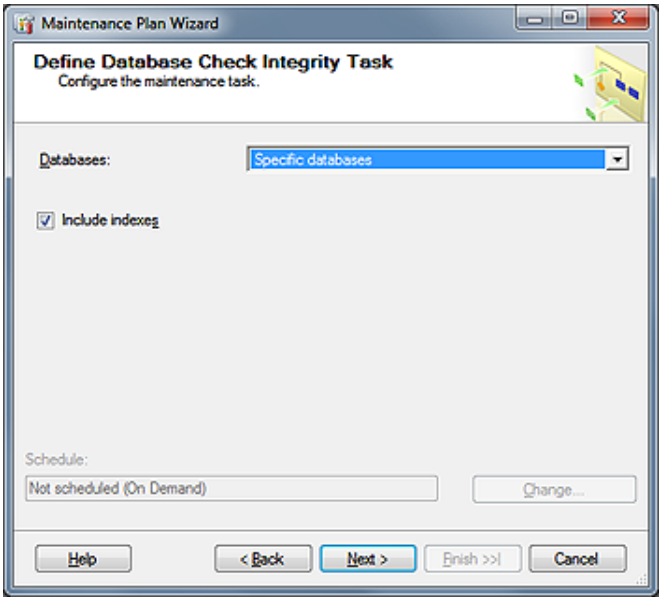


Figure 13-20

As you can see in Figure 13-21 which was instantiated by clicking on the down arrow at the end of the **<Select one or more>** option, the default setting is: ◉ **These databases:** where one would actually check the database(s) to include in this plan. As I mentioned previously, this setting **is not** sticky, and you will need to reselect the database(s) on each of the successive dialog forms in this Wizard where you come across the verbiage **Database: <Select one or more>**.

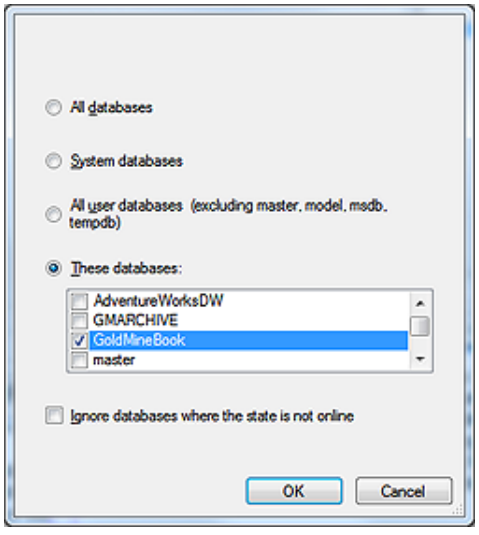


Figure 13-21

I have only selected a single **GoldMineBook** database, however, you could have selected all of your GoldMine databases if you utilize more than one (not recommended). One would next click on the **OK** button to return to the **Define Database Integrity Check Task** dialog form.

Step 08

Clicking on the **Next >** button will bring you to the **Define Reorganize Index Task** dialog form displayed in the next column.

Again, this is not the default dialog form as I have already selected **Specific databases**, although everything else on this dialog form remains in its default state.

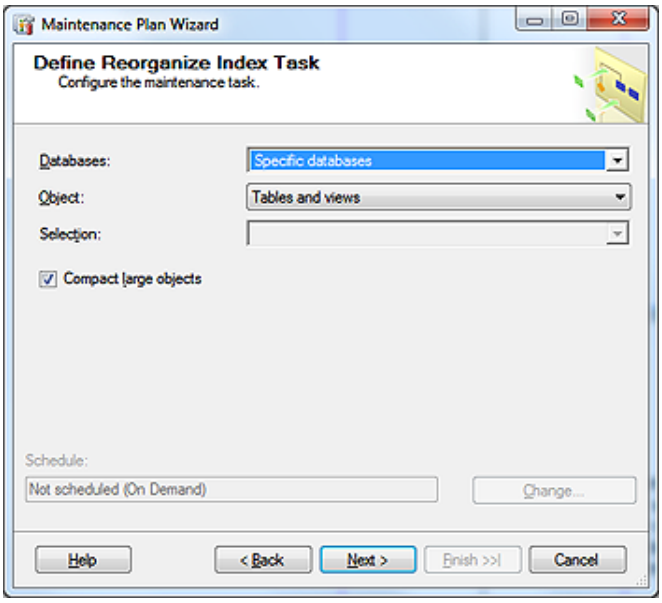


Figure 13-22

Step 09

Clicking on the **Next >** button will bring you to the **Define Rebuild Index Task** dialog form displayed here in Figure 13-23.

Again, this is not the default dialog form as I have already selected **Specific databases**, although everything else on this dialog form remains in its default state.

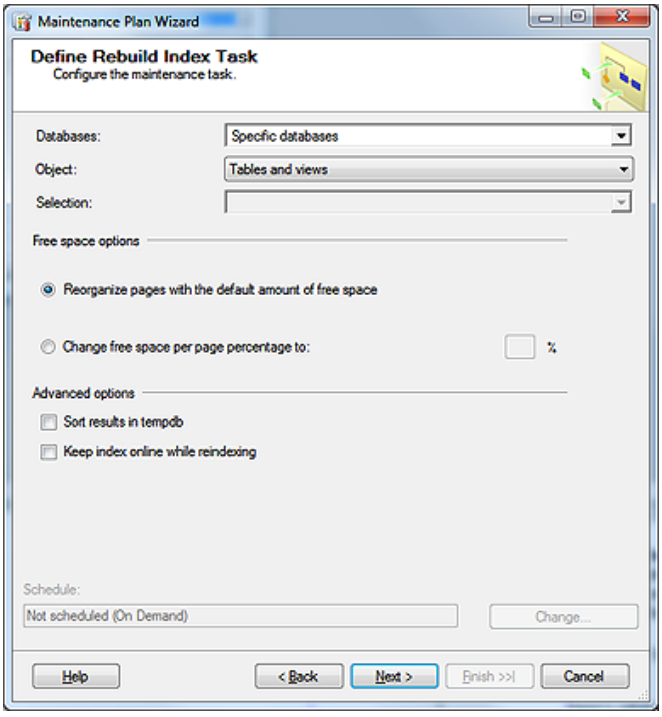


Figure 13-23

Step 10

Clicking on the **Next >** button will bring you to the **Define Update Statistic Task** dialog form.

Again, this is not the default dialog form as I have already selected **Specific databases**, although everything else on this dialog form remains in its default state.

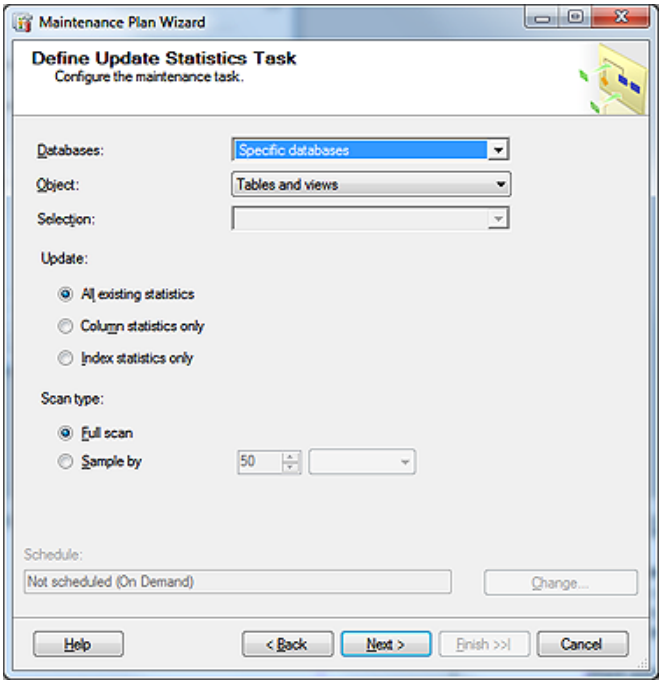


Figure 13-24

Step 11

Clicking on the **Next >** button will bring you to the **Define History Cleanup Task** dialog form show at the top of page 9. Everything on this dialog form remains in its default state.

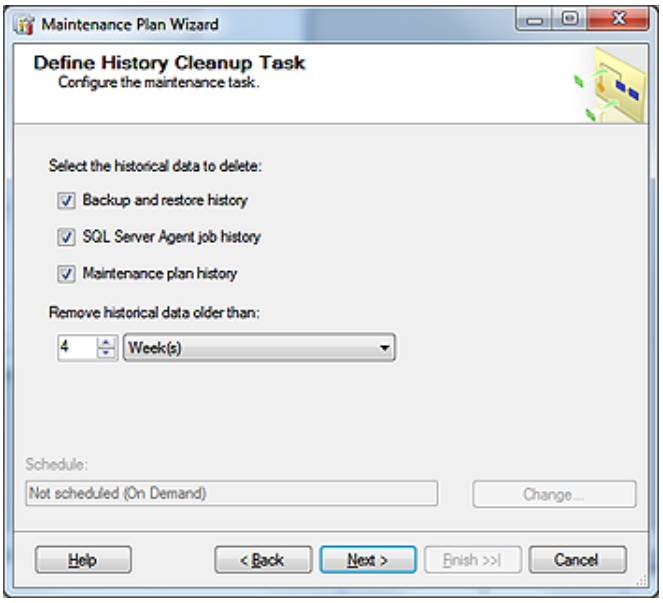


Figure 13-25

Step 12

Clicking on the **Next>** button will bring you to the **Define Back Up Database (Full) Task** dialog form displayed in the next column.

Again, this is not the default dialog form as I have already selected **Specific databases**. Unlike the other screenshots, I have made changes to the default settings on this dialog form.

For instance, I did select to ☑ **Create a sub-directory for each database**, and, as well, I did select to ☑ **Verify backup integrity**.

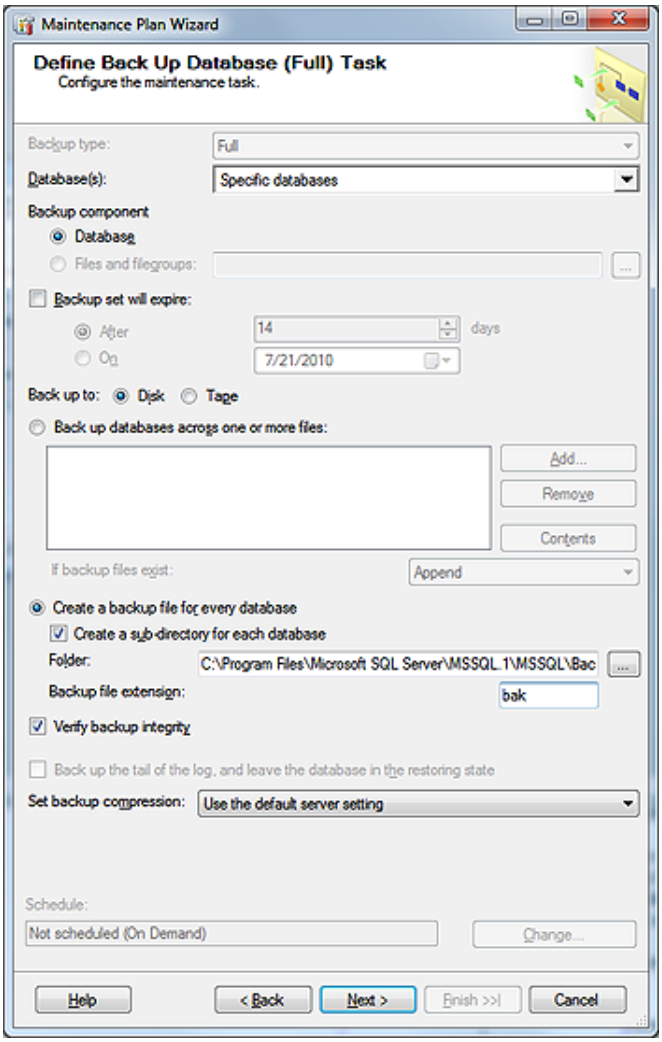


Figure 13-26

Step 13

Clicking on the **Next >** button will bring you to the **Define Maintenance Cleanup Task** dialog form shown here in Figure 13-27.

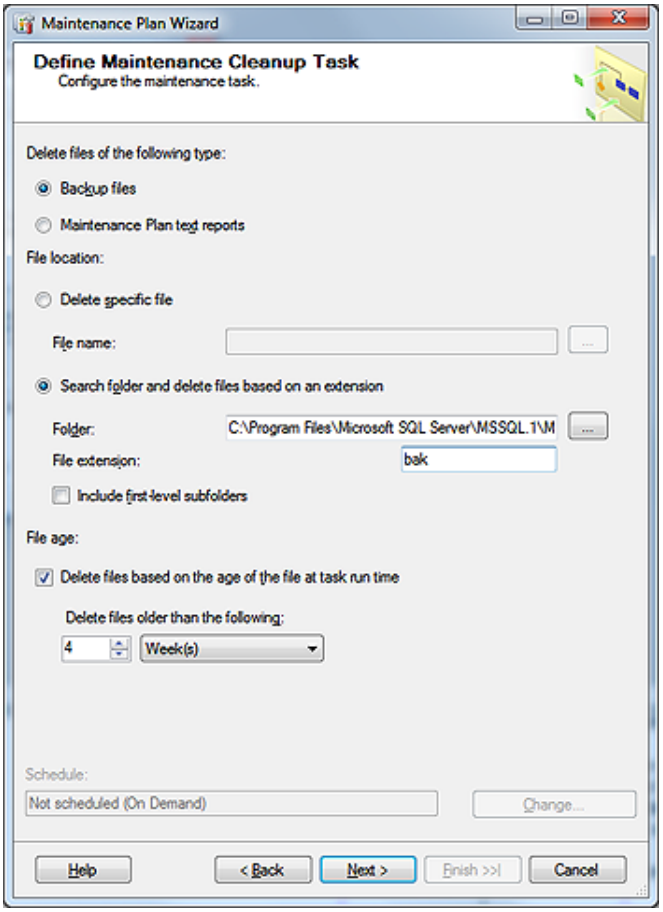


Figure 13-27

In the default state for this dialog form the ⊙ **Search folder and delete files based on an extension** is selected, however, it is up to you to supply the actual **Folder:** in which to search and the **File extension:** for which you wish to search.

In my particular case I used the ellipsis (**...**) button to browse to my designated backup folder and the selection entered:

C:\Program Files\Microsoft SQL Server\MSSQL.1\MSSQL\Backup\SQLGoldMine

I also entered the file extension of **bak** which is the default extension as accepted on the above **Define Back Up Database (Full) Task**.

Step 14

Clicking on the **Next >** button will bring you to the **Select Report Options** dialog form as displayed in Figure 13-28.

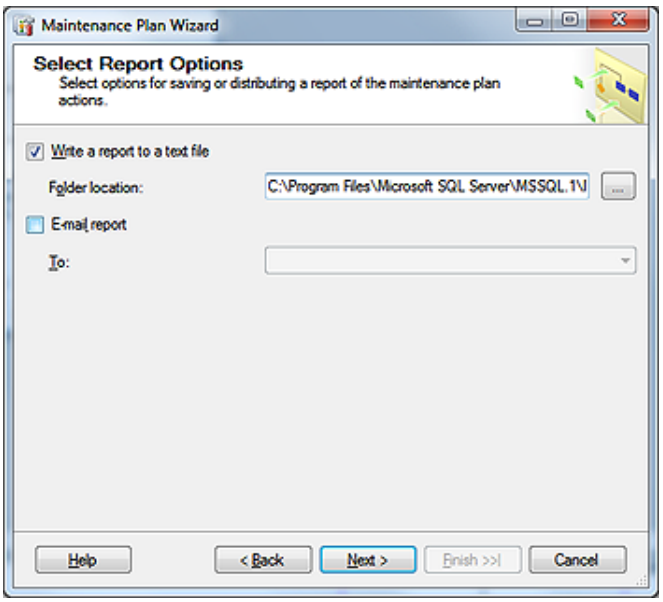


Figure 13-28

I usually just accept the defaults here, however, I have had instances where the clients had wanted to also have the report e-mailed to them. In those cases, I would have to select the ☐ **Email report** option while including an e-mail address in the **To:** field.

Step 15

Clicking on the **Next >** button will bring you to the **Complete the Wizard** dialog form

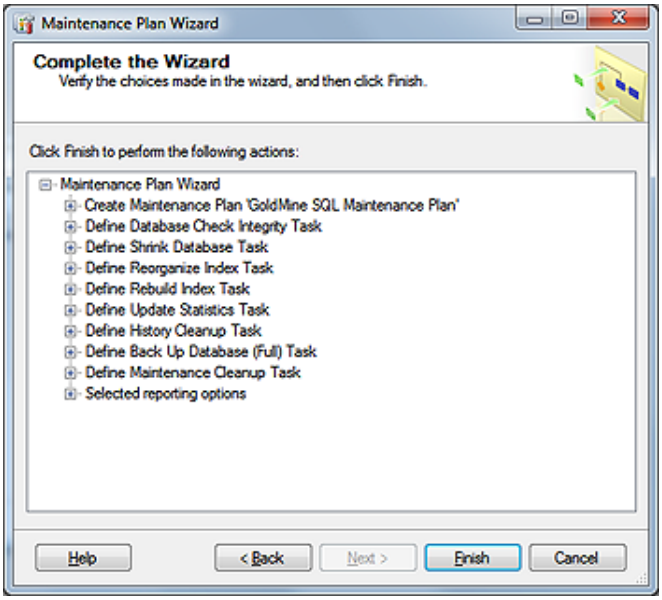


Figure 13-29

Finally, we have completed the Wizard, and we have little more to do than to click on the **Finish** button which will generate the **Maintenance Plan Wizard Progress** dialog form, refer to Figure 13-30. Hopefully, yours will also display the **Success** check mark. Clicking on the **Close** button, and you will have created your GoldMine SQL Maintenance Plan.

Step 16

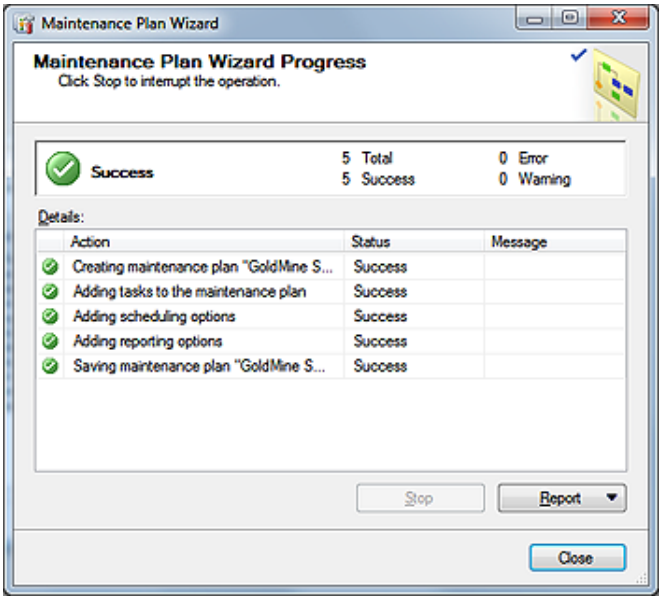


Figure 13-30

3 days later you decide to check your **Job Activity Monitor**, and you notice that your GoldMine SQL Maintenance Plan has never run. You look in your SQL Backup folder and there are no backups to be found.

Why do I even bring this up, because just yesterday I had a client call me to tell me that the plan that I had created for them wasn't working, and hadn't worked since June 30th, 2010.

My first thought was to open SQL Server Management Studio to see if there was anything obviously wrong. Sure enough, there it was staring me right in the face. The **SQL Server Agent** had been **Stopped** as often happens when one stops the **SQL Service** itself. People remember to turn the **SQL Service** back on, but often forget that the **SQL Server Agent**, which is automatically stopped when the **SQL Service** is stopped, **must be Started**.

The dialog form shown in Figure 13-31 on the next page has the **SQL Server Agent** highlighted, and the icon to the left displays a green right facing arrow indicating that the service is **Started**. It is imperative that this service be started if you want any of your SQL Maintenance Plans to function at all.

Once I restarted the service for my client, the GoldMine Maintenance Plan ran Successfully again.

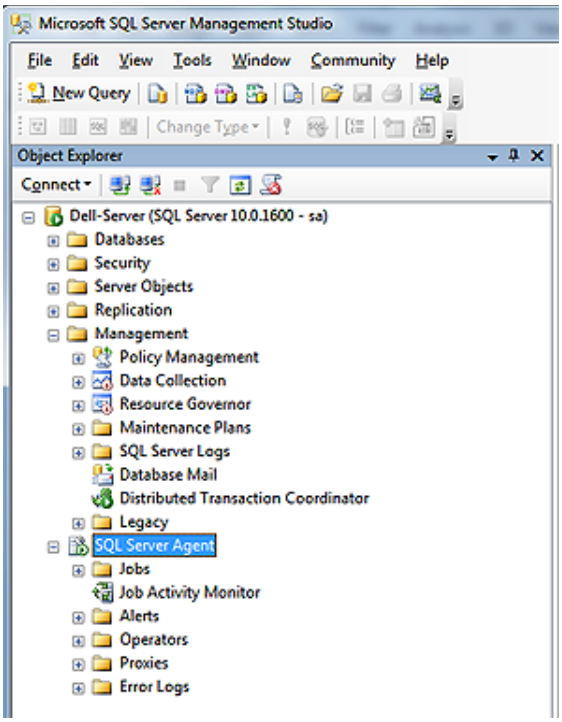


Figure 13-31

Conclusion

Caveat

Please remember that anything that you do within the **Microsoft SQL Management Studio** is **not** sync aware, hence, the transactions **will not** be propagated out to your Remote-side Undocked/Site License Users.

This means that any changes that you make through the Microsoft SQL Management Studio Server-side, will have to be reproduced by hand through the Microsoft SQL Management Studio on any of your Remote-side installations.

This pretty much wraps up this chapter for us, however, I would like to mention some of the powerful features that are accessible to you via the **Microsoft SQL Server Management Studio**.

You may **Open** any of your the tables within your database, and manipulate the data any way that you wish. You may even **Delete** records in this table once it has been opened.

You may script any table within your database using the **Script Table as** feature. This will allow you to build: **CREATE to**; **DROP to**; **SELECT to**; **INSERT to**; **UPDATE to** & **DELETE to** scripts which can be saved and utilized anywhere.

You can also **Add Fields**, **Add Indexes**, and do just about anything at all that you want within the Microsoft SQL Server Management Studio.

14

In This Chapter

After writing the original The Hackers Guide to GoldMine books, I found that a lot of my readers really appreciated the content, and the way that I handled the subject matter. However, they all, and I do mean all, wanted more. "Could you cover Reports?" "Could you cover GoldSync?"

I felt that GoldSync is an often maligned subject, and that there is little good documentation on the subject. I, therefore, decided to include this chapter in The Hacker's Guide to GoldMine series of books after the first book in this series, and continuing that line, in The Definitive Guide series of books.

There has been a few changes over the years, however, there has been little change in GoldSync between GoldMine 7 Corporate Edition, and GoldMine Premium other than possibly menu location, and performance fixes.

I will begin my discussion with the Synchronization Settings, which is actually a GUI for the GoldSync.ini. I then plan to cover the Server-side set up of GoldSync, the IP Address, and ending my portion of this chapter with the Remote-side set up of GoldSync. Finally, I have been able to add, thanks to David Lee the author, a Whitepaper on GoldSync which you GoldSync Administrators should find to be an eye opener.

GoldSync.ini

GoldMine supplies a GUI interface to some of the GoldSync.ini settings. This can be found by users possessing Master Rights, when they select:

Tools
Synchronize ►
Synchronization Settings...

...from the GoldMine menu. This will bring up the dialog form shown here in Figure 14-1:

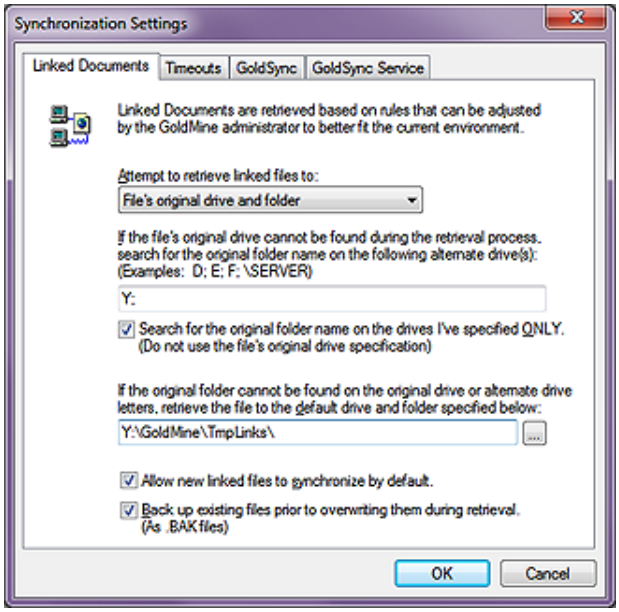


Figure 14-1

Linked Documents

On initialization, the GUI positions you on the **Linked Documents** tab, and it is here that one sets their options for the GoldSync actions with regards to Linked Documents retrieved during the synchronization process. The key word here is retrieved. GoldSync wants to know what rules to enforce when retrieving a linked documents, and remember, that these rules must be established **Server-side**, and **Remote-side** alike. They are specific to side that is being set up.

Note

In older versions of GoldMine this GUI was found under the user Preferences, Sync tab. As of GoldMine Premium, it has been moved to this, a more corporate accessible location.

Tip

The Server-side GoldMine installation should always be via a mapped drive letter resulting in a path similar to:

Y:\GoldMine\...

Whereas the Remote-side GoldMine installation should always be to the C:\ or D:\ drive letter resulting in a path similar to:

C:\GoldMine\...

Tip

I always recommend that you create a folder under the GoldMine folder, and then train your end users to store all of their linked documents under that folder, both those sent and those retrieved.

Let GoldMine be your document manager.

If you follow this philosophy, then it is extremely important that you maintain this pathing, less the drive letter, on all of the Remote-sides as well.

Note

The settings displayed in Figure 14-1 apply to the Server-side on my system, but all of the Remote-sides must be configured as well. Usually the only differences between the two sides are the drive mappings.

Let's look at the first rule under this tab. **Attempt to retrieve linked files to:** is a drop list with but two choices. The default choice is: **File's original drive and folder**, while the other option is: **Specified default drive and folder**. As preparation for this choice, I suggest that you read the Tip in the sidebar. I would suggest, unless you have reason to do otherwise, that you stick with the default selection in this case.

If you followed my tip, then the next option is where one would establish a rule that will compensate for the drive letter differences in between the Server-side path and the Remote-side path. For Instance; your Remote-side might save a linked document in: **C:\GoldMine\Documents**, while the comparable path on the Server-side might be: **Y:\GoldMine\Documents**. Since the file's original path, including drive letter, does not exist on the server, it is important that one suggests other possible drive letters. As I never recommend using UNC paths, \\Server\C\, I do not suggest that you incorporate any of these in this rule. All of the optional drives to be searched should be separated with a semicolon (**;**). Based upon my For Instance, I would want to include **Y** in this space on the Server-side configuration, while I would have used a **C** on the Remote-side configuration.

As you may know, based on this configuration, that the linked document drive location will never be the same as the senders linked document drive location, you may want to reduce some search time by selecting ☐ **Search for the original folder name on the drives I've specified ONLY. (Do not use the file's original drive specification)**. This will force GoldMine to not search in the path of the documents original drive letter, bypassing one step.

Next, one must instruct GoldMine as to where the linked document files should be placed if, and when GoldMine can not find a place to put the document based on the afore mentioned rules. The default setting is always under the GoldMine folder, in a folder called **TmpLinks**. There should be no reason for you, Server-side or Remote-side to adjust this option.

The next option is interesting. By default, it is selected. ☑ **Allow new linked files to synchronize by default**. In the days of yore, one had to make sure that they selected to allow the file to synchronize when they were creating the Linked Document, and there were many times when the user would forget to select that option. Of course, back then, dial-up Internet was the fastest way to synchronize, and one may not have wanted to synchronize documents in an attempt to keep the transfer set size to a minimum. Today, with high speed Internet, this is a rather moot point, hence the reason that this option was added here as an override feature to reinstate the default behavior.

While the next option, ☑ **Back up existing files prior to overwriting them during retrieval**. **(As .BAK files)**, is also selected by default. Personally, I do not feel that this is necessary as there is a back up already at the transfer set senders location. As always, I leave the choice up to your discretion. At the cost of hard drive space today, storage is not an issue, so what would it hurt to leave the default setting?

Any changes made under this tab are reflected in the GoldSync.ini thusly:

```
[LDRetrieval]
   Option=0
   Backup=1
   UseAltDrivesOnly=1
   AltDrives=Y
   TmpLinksPath=Y:\GoldMine\TmpLinks\
```

Timeouts

This tab is a legacy hold over, and in point of fact, some of the settings contained in the GoldSync.ini can only be accessed by editing the GoldSync.ini directly. As the verbiage on the dialog form states, *"The options below allow you to override the default handshake timeouts of each synchronization method. There is usually no reason to change the defaults."* As you might have surmised, the statement hasn't been changed, it still states *"...each synchronization method."*, but the dialog form only shows one synchronization method, the **Internet Connection Method**. I will discuss the GUI first, shown in Figure 14-2 on the next page.

The **Handshake timeout (sec):** is set by default to **20** seconds, while the **Connection timeout (sec):** is set to **60** seconds by default. I have chosen not to accept the defaults, and I am actually utilizing **60**, and **120** respectively. Why? It has always worked in the past so why change it if it hasn't failed. In other words, it was set that way when I was synchronizing via a dial-up connection, and as I upgraded GoldMine through the years, the settings have gone right along with the upgrade. Once changed, they could be reset to their default state by simply clicking upon the **Set Defaults** button. One might want to, or need to change these settings if they had slow Internet connectivity. These days, in corporations, that is relatively unheard of. Yet GoldMine sells to the masses, and many of its users are 1, 3, 5, and 10 license users that have a tendency to have the cheapest, slowest Internet access. Hence, the need for FrontRange to maintain this GUI.

I have had the need to double these settings for a couple of my clients. In a couple of cases only, I have set these to 40, and 120 respectively.

Any changes would affect this section of the GoldSync.ini:

```
[GoldSync]
ModemTimeOut=10
WanTimeOut=20
WanRereadTime=100
mInetConnTimeOut=120
mInetHSTimeOut=60
```

I have included some of the legacy items in this section, however, only those in dark blue are affected by the Timeouts tab via the GUI. The others, if they are to be changed at all, must be changed by hand by modifying the GoldSync.ini using NotePad.exe. Again, if you must change any of the GoldMine ini's, I always recommend that you do so using the NotePad.exe application to ward off any risk of formatted characters being included in the ini.

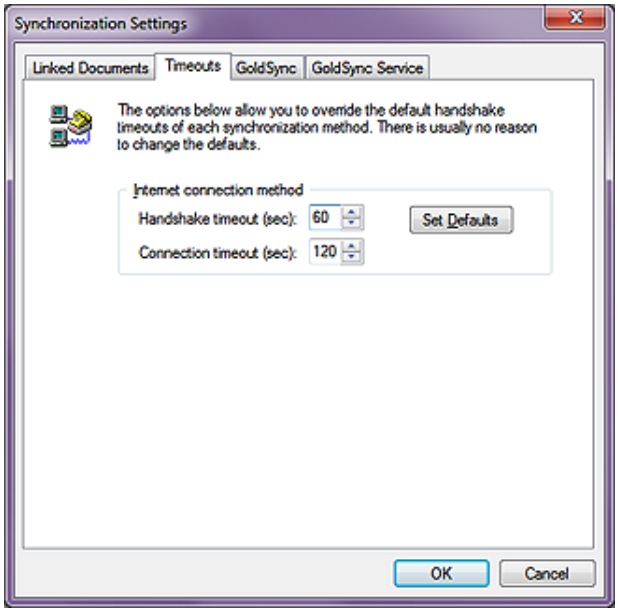


Figure 14-2

You will notice that the first statement is for modem handshake, while the next two are for Wide Area Network (WAN) handshake, and connection. I have seen the need to change the WAN settings occasionally, but it has been a long, long time since I have seen anyone change the modem setting.

GoldSync

With respect to the life of GoldMine, the GoldSync tab, Figure 14-3, is relatively new, and allows one to automatically have an e-mail sent to a given e-mail address if, and when a particular synchronization session fails.

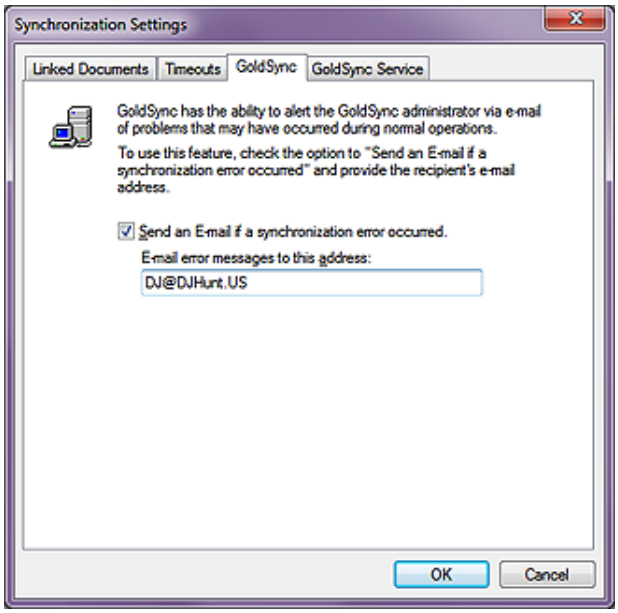


Figure 14-3

This is a relatively simple dialog. One must check the box, and one must add an e-mail address, if they desire this feature. Doing so could result in these GoldSync.ini settings:

```
[GoldSync]
EmailErrors=1
EmailAddr=DJ@DJHunt.US
```

GoldSync Service

WARNING

At this writing, I can still not recommend the use of the GoldSync Service. Many users find its stability, even today, to be iffy on the best of days.

That was pretty much all that there was to the GoldSync.ini GUI in GoldMine versions prior to 6.50.31113, however, with the introduction of newer versions came a new tab, **GoldSync Service**, refer to Figure 14-4 on the next page. This had been an often requested addition to GoldMine, the ability to run GoldSync as a Service. This tab offers one of the couple of ways of installing GoldSync as a Service. The Administrator simply clicks upon the **Set up GoldSync as a Service** button, and that's it. Once established, this button will be disabled, while the other two buttons will become enabled. Once the service is established, you may select to **Configure GoldSync Service** or to **Uninstall GoldSync Service**, neither of which requires a brain surgeon to manage. One may also **Start**, and **Stop** the **GoldSync Service** from the **GoldSync Administration Center** discussed later in this chapter.

You should remember that it is important that the **GoldSync.ini** be setup **Server-side**, and **Remote-side** with settings that are appropriate to the side on which you are working. For example on the Server-side the local drive to search for documents might be the **Y** drive, while on the Remote-side that same entry might be the **C** drive. If documents aren't moving back, and forth smoothly, this is one of the first places that I look for fault.

GoldSync Server-side

Note

In setting up the Server-side for GoldSync, one would be offered the opportunity to create an undocked license if an available undocked license could not be found.

After years of practice, however, I have found that my system works. In this book, therefore, I will convey my system, and you may modify that according to your needs and desires.

Note

You are advised that these steps should be performed on the system that will be acting as your GoldSync Server. That way, when we configure the GoldSync Server itself, it will be configured for the correct system.

You may, however, set up different servers, if you so choose, after you have the first setup working properly.

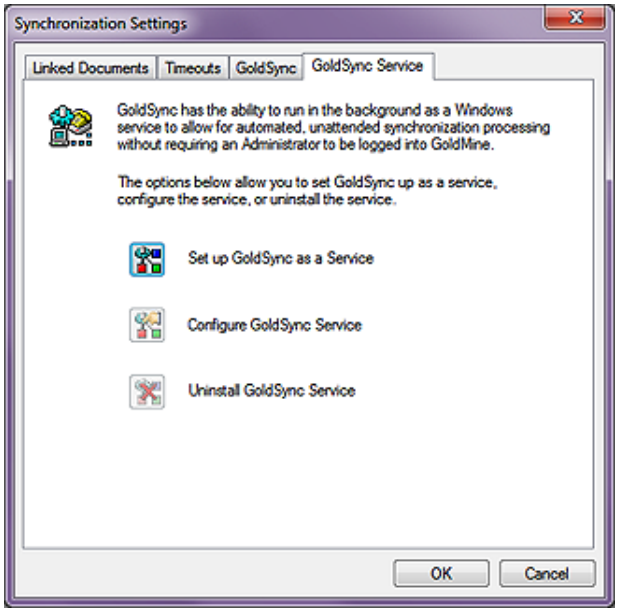


Figure 14-4

Let's begin to look at the **GoldSync Server-Side** setups, but before we begin that, I am going to assume that I will have at least one **Remote-side** install. Based on that assumption, I will first want you to create an **Undocked User License** or a **Site License** via the **License Manager**. The **License Manager** was covered earlier back in **Chapter 2** so I will not be reexamining it here. If you are following along, please create an Undocked License or Site License for a user or site now, in preparation for our Server-side setup.

Again, from the GoldMine menu:

Tools
Synchronize ►
Synchronization Settings...

This should, if you haven't set up anything here in the past, bring up a message box as shown here in Figure 14-5, which I have already accomplished so I had to copy this image from a previous book.

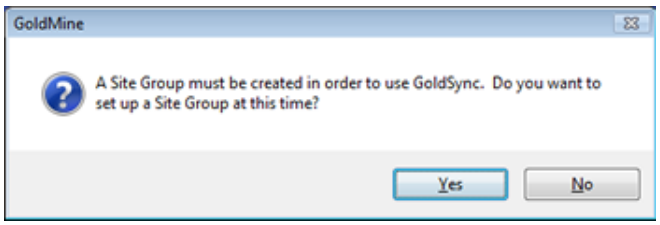


Figure 14-5

I want to walk you through this by hand, such that I have selected the **No** button on this message box. This action results in the display form shown in Figure 14-6. Of course yours won't have anything under the **Site Groups** or **Servers** if this is the first time that you are in this dialog form. From here, the first thing that I will do is to create a **Site Group**. These are the remote sites that will be synchronizing with your corporate GoldMine. These could either be Undocked Licenses or Site Licenses, it does not matter.

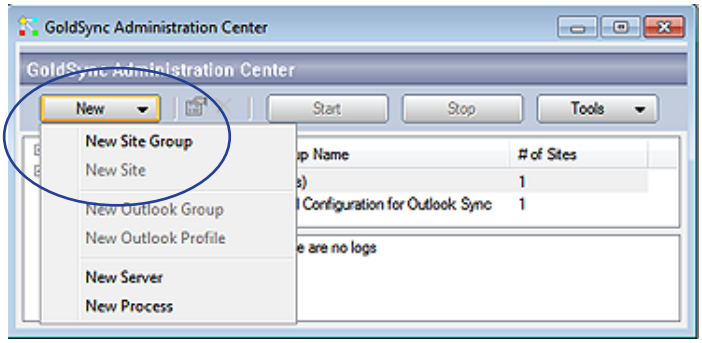


Figure 14-6

Note

Why have multiple Site Groups?

Each Site Group can have its own synchronization settings, and the settings are inherited by the remotes within that Site Group unless you specifically override a particular remotes inherited settings.

Instead of having to do each remote site individually, it is sometimes easier to simply add them to an existing group that has similar settings.

Figure 14-7

To begin this task, I can either click on the **New Site Group** menu item (encircled) or I can right-click the **Site Groups** tree, and select **New...** ► from the local menu, and then again on **New Site Group**. Either selection begins the Wizard, which will walk one through the creation process, the first page of which is shown here in Figure 14-7.

This dialog page is simply asking for a colloquial name that you wish to use to describe this particular site group. If you plan to utilize

multiple site groups, then you want to make certain that these descriptive names are, in fact, descriptive. I have entered **Remote User(s)** for this my first site group, and then clicked on the **Next >** button.

This action would then bring us to the next dialog page as shown Figure 14-8. You will notice that there are two separate frames, however, they both contain radio button selections. Even though they are dispersed between two frames, one may only choose a single radio button from anywhere on this dialog form. For this exercise, the frame that I am interested in is the **Connected Methods (IP to IP/Network)**. The other frame is for **Non-Connected Methods (Internet E-mail, Shared Directories)**. In the former frame, I have two methods from which I may choose:

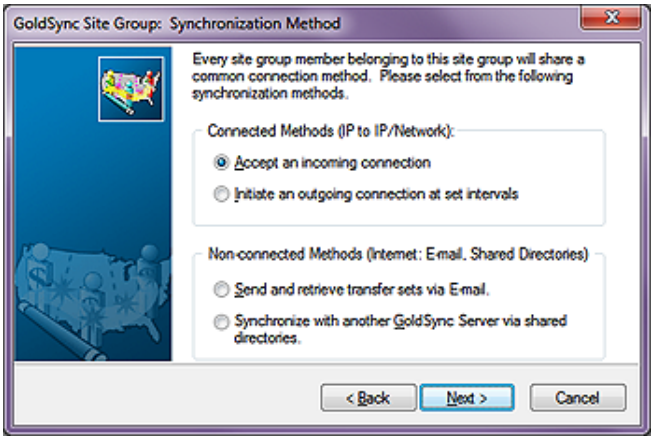


Figure 14-8

- ◉ **Accept an incoming connection**
- ○ **Initiate an outgoing connection at set intervals**

While the latter frame, which also has two methods, contains:

- ○ **Send and retrieve transfer sets via E-mail.**
- ○ **Synchronize with another GoldSync Server via shared directories.**

I have accepted the default of ◉ **Accept an incoming connection** from this **Site Group**, and then I go ahead and clicked upon the **Next >** button to move on to the next dialog page of the Wizard as depicted in Figure 14-9.

Your first two options are usually selected as the default state of this Wizard, and I would recommend that you keep them selected.

- ☑ **Send changed data to site group's members**
- ☑ **Retrieve changed data from site group's members**

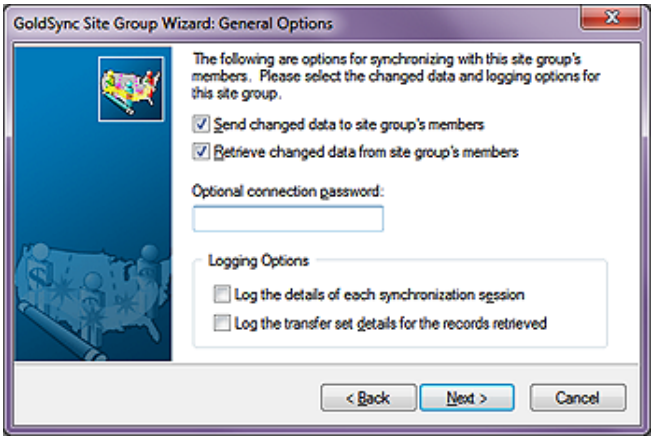


Figure 14-9

This is your first real indication that you are not setting up a sync session for an individual, but, instead, that you are setting up the sync session for a group of individuals. Whatever you set on these pages will be the default setting for the Server-side for all sync sessions, unless they are specifically overridden on a site by site basis.

With these two options selected, I am stating that the Server-side is willing to send, and retrieve information from any sites associated with this site group.

I am next presented with the option to supply an **Optional connection password:**, and I decline this option. If you are really super security conscious, you may, indeed, want to supply a password here. Should you, however, you must remember to set the exact same password on the Remote-side setup or the synchronization process will fail.

Next I have the frame for the **Log Options**. Selecting these options will cause the GSLogs table to become extremely large over time. Personally, I usually will select the first option, and not the second option. I later refine this if the syncs are going well, or badly as is appropriate. If they are going well, I will deselect the first option, while if they are going badly, I may select the second option.

Note

Rule of Thumb:

Will you understand what is contained in this site group if you haven't seen the GoldSync Administrator for over a year?

You do this because GoldSync offers you the ability to add multiple Site Groups.

Tip

It is important that you position your frame of mind properly when walking through the Wizard.

I am currently setting up the Server-side, hence, my settings, for what I am sending, are based on what the Server-side is sending to the Site Group, as the retrieve options are what the Server-side is willing to retrieve from the Site Group.

When I get to setting up the Remote-side, the reverse is true. From that position, I will be setting up GoldSync to say what the Remote-side will send or accept from, and to the Server-side.

Additionally, it is important that you understand that just because the Server-side is sending the information, if the Remote-side is not equally set to retrieve that information, then the information effectively goes no where.

Understanding your position with respect to Server-side/Remote-side is extremely important.

☐ Log the details of each synchronization session
☐ Log the transfer set details for the records retrieved

The first option only logs the main details. Was the session successful? How many Contact Updates or New records were added? While the second option logs the minutia of detail that is happening on a record by record basis. I think that you can see that if you had a number of remotes synchronizing daily, your TLogs, when this option is selected, could become extremely large very quickly, and you should be constantly purging your TLogs (hopefully). That reminds me, I haven't Purged my TLogs in some time.

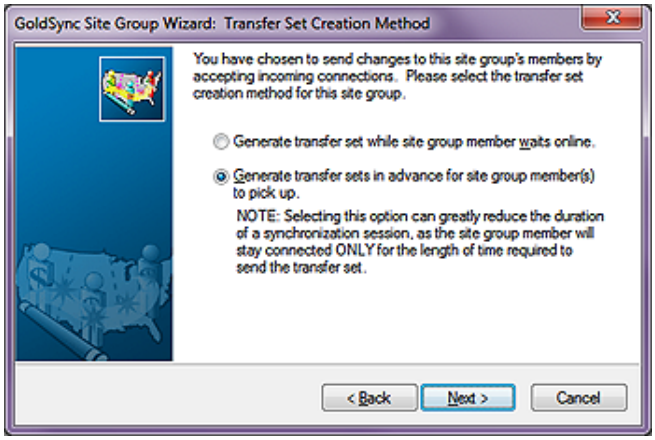


Figure 14-10

Again, I select the first option only at this point, and then I click upon the **Next >** button to bring me to the next dialog page, the **Transfer Set Creation Method** page of the wizard, shown here in Figure 14-10.

This dialog page is what I call a legacy hold over. With the advent of high speed Internet, and high speed networks, the default option to ◉ **Generate transfer sets in advance for site group member(s) to pick up.** is not as necessary as it was in the past, where dial-up networking was relatively slow. In fact, I have not selected this option for my clients in over 4 years, and I probably never will need to select this option again.

I always select to ○ **Generate transfer set while site group member waits online.** over the default option given todays environments. I would then click on the **Next >** button again to move along to the next dialog page. Refer to the **Send Options** page of the wizard, Figure 14-11, to continue following along with my ramblings. Again, remembering our position, these settings will pertain to what the Server-side wishes to send to the Remote-side recipients.

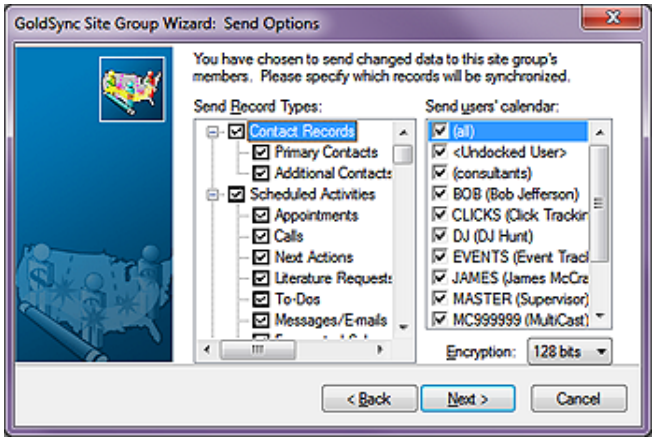


Figure 14-11

For this exercise, I will make the assumption that my remote recipients, as a group, will receive everything that the GoldMine Server-side has to give out. By default all of these items are selected, but to assure myself that this is the case, I right-click over the **Send Record Types:** list to bring up the local menu. From the resulting local menu, I will choose **Select All**.

I then right-click over the **Send users' calendar:** list, and again, I choose **Select All** from the resulting local menu. This assures me that the Server-side will be sending everything for everybody to each of the remote recipients in this group.

As always, when I am setting up the Remote-side, I could select to not accept everything that the Server-side has offered in the transfer set. Both sides, respectively, must decide what they wish to send, and what they wish to receive from each other. As this is a Group setting, you should understand that you can always override these settings on an individual basis. I can tell you here and now that I never send everything from the Server-side to the Remote-side. I make good use of the individual overrides.

There is an additional entry, on Figure 14-11, that I should discuss, **Encryption:**. You have but two options, the default **128 bits**, or **32 bits**. This is also more of a legacy hold over from when we were not allowed, by law, to send encryption routines out of the United States that were higher than 32 bits. Today, of course, that no longer matters, and most users will just accept the default 128 bits option.

Tip

If synchronization is up-to-date for all of your Remote-side users, I always recommend that you Purge your TLogs on a daily basis. Remember that this is two-sided activity again, and purging must be done on the Remote-side as well as the Server-side.

Note

David Lee, Whitepaper author, points out that ◉ Generate transfer sets in advance for site group member(s) to pick up speed has little to do with connection speed. Rather, it has to do with the time required to build the transfer set itself if in a large user environment. On larger installs with large numbers of users and/or large databases, you could get a big traffic jam. Queued processing could be very helpful in this scenario.

Note

I would not normally do this when I can see all of the Users/User Groups in my list. Simply selecting ☑ (all) should suffice.

Tip

I actually have two databases configured. For years I have espoused that you should only have one database with two exceptions.

If you purchase a mailing list, you may want to put it into a separate GoldMine Contact Set until you have had time to qualify the names. At that time, you would move the qualified names into your working GoldMine Contact Set.

The second scenario would be when you are Importing records into GoldMine. It is sometimes wiser to Import them into a separate Contact Set, than into your working Contact Set. Once, you have ascertained that the Import was as you desired, you could then move those records into your working Contact Set.

David Lee points out two other approved reasons for additional databases:

A. A training database
B. An Archive database

Tip

*The **File Code** for a given database **must** be the same on both the Server-side and the Remote-side.*

Click the **Next >** button, and move on to Figure 14-12, **Send Contact-related Options**. From this dialog form I have a drop list which provides three options from which to choose, and I have my databases from which I may desire to send information. I will elaborate more on this in a minute.

Let's look at the first option that one must make a decision about.

The items that one has from which to choose are:

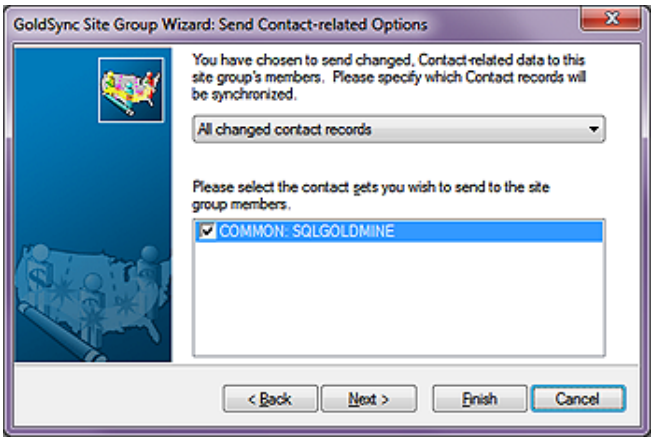


Figure 14-12

All changed contact records
Contact records linked to the 'Send user's calendar' list
All filtered records and user-scheduled activities' records

For this exercise, for these site groups, I choose to send **All changed contact records**. There may be cases where one might want to send just a filtered set of contact information to the Remote-side. That being the case, I suggest that you do not set it here for the site group in general, but instead, with an override which I will discuss later in this chapter.

Next, the GoldMine Wizard asks us to **Please select the contact sets you wish to send to the site group members.**. As this implies, one could send deltas from one database or from many databases. I almost want to call this a legacy hold over as well, as I advocate that you should have only one working Contact Set, and certainly, by my paradigm, you would not want to synchronize more than one database. This is especially true today with the enhancements to **Record Typing** that we saw emerging back in GoldMine 6.5. However, all that said, this section provides one with the ability to send, and later retrieve, from more than one contact dataset. For this exercise, as you can plainly see in Figure 14-12, I have selected ☑ **COMMON: SQLGOLDMINE**. Please note that the alias COMMON: represents a unique File Code which you are required to enter for all newly created databases. I used COMMON, whereas you could name yours anything unique to your datasets.

Click on the **Next >** button to move yourself to the next dialog page of the Wizard, shown here in Figure 14-13.

This is the **Send Filter Options** page of the Wizard. From this drop list, you would be able to select the default **All Contact Records!**, or from the **< Filters of: DJ (DJ Hunt)... >** which means you could utilize any users filters that exist in your GoldMine database. Again, as I am establishing the Site Group settings, I will select the default, and override the individual sites for filtering if and when necessary.

Figure 14-13

Let's click on that **Next >** button as shown on Figure 14-13 to move ahead to Figure 14-14 which can be view by the reader on the next page.

As you see in Figure 14-14, on the next page, I now have the ability to define what the Server-side is willing to accept (Retrieve) from the Remote-side users who are part of this Site Group. On this, the **Retrieve Options** dialog page of the Wizard, and as this is the retrieve options for all users in this Site Group, I will right-click in the **Retrieve record types:** list, and I will choose **Select All** from the Local Menu. Under the **Retrieve user's calendar:** list, I will right-click, and I will choose **Select All** just as I had done for the Send Options for the Site Group.

Tip

*Even though I have asked you to **Select All**, you may want to consider turning off, unchecking, the **Logs**. These can grow abnormally large if not purged regularly, and are not necessary to retrieve from the Remote-side unless you are experiencing issues with synchronization, and you wish to analyze the **TLogs**.*

Tip

David Lee says: "You might want to turn off the ability to accept record deletions." This will protect you from some errors from the Remote-side(s). Instead, develop an SOP where they flag the records as Delete or Archive, and then let the System Administrator globally delete or archive them as appropriate for the needs of the corporation. I recommend this design in almost all large synchronization environments.

WARNING

*As of my book The Hacker's Guide to GoldMine Premium, you could not use the **Add Other Licenses** option. After a long discussion with FrontRange, FrontRange stated that this option was never meant to work as it had in the past, and that they had corrected it.*

*In my opinion, it had always worked properly in the past, and what FrontRange considers to be **"corrected"** in GoldMine Premium, is to not have it function at all.*

Am I missing the big picture here?

Either way, I have not tested it to see if FrontRange has gone back to the old functionality in the latest GoldMine Premium 9.0.1.49.

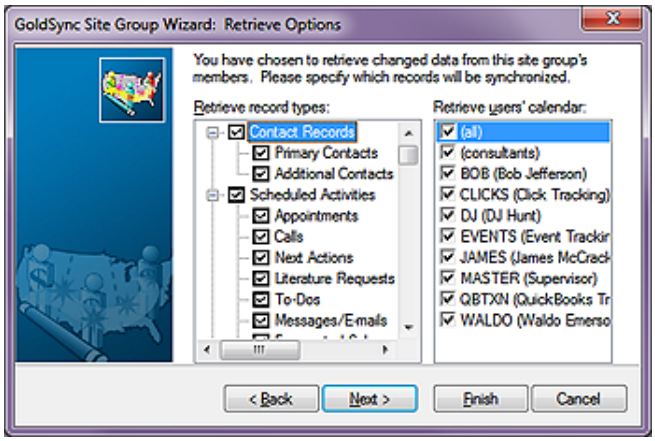


Figure 14-14

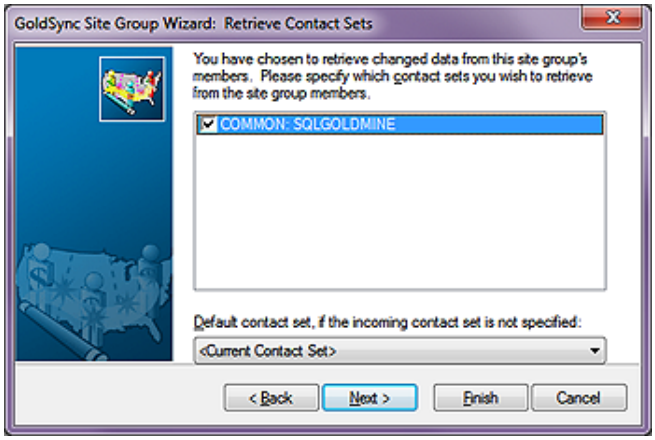


Figure 14-15

As before, if this is your first time setting up a Site Group, these items should be all selected by default, however, to be safe, I opt for making the selections myself. I trust myself more than I trust the GoldMine designers programming capabilities.

Again, I will click on the **Next >** button to move forward.

I have now advanced to the **Retrieve Contact Sets** page of the Wizard. If you have more than one contact set established, you must select those contact sets that you are willing to retrieve for the Site Group as a whole. For the Site Group, I am willing to retrieve all of the **COMMON: SQLGOLDMINE** contact set, as I can override this on the individual site later on if I find that necessary. You should notice that each dataset is preceded by its unique **File Code**.

The **Default contact set, if the incoming contact set is not specified:** option allows us to select from any of your defined contact sets. Usually, however, I opt for the default setting which is to use the **<Current Contact Set>** as I only recommend the use of one database in our paradigm, and, again, I click on the **Next >** button to advance to the next Wizard page.

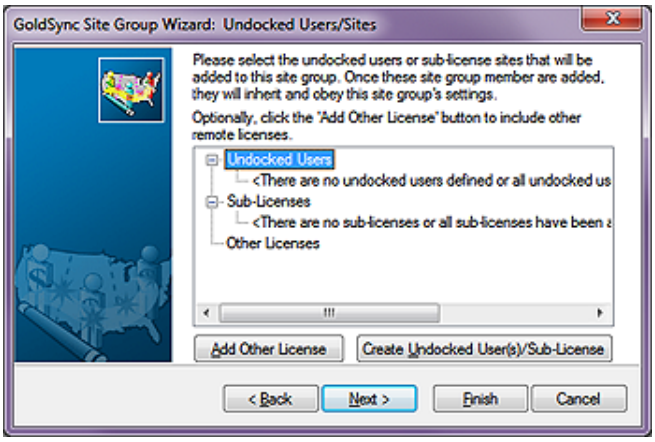


Figure 14-16

This is the page, Figure 14-16, from which I could select the **Undocked Users/Sites** that will be included in this Site Group assuming that you had preset Undocked/Sub-Licenses Licenses. You may select as many or as few as you wish, but you must have at least one selected for the Site Group or all will be for not.

I had three that were predefined via the License Manager as Undocked Users. However, these had been already added to a previously defined Site Group so that there are no more currently available for this Site Group. As you can see, you could add undocked user(s) or sub-licenses on the fly by clicking on the **Create Undocked User(s)/Sub-License** button. One could also simply use any other GoldMine license via selecting the **Add Other Licenses** option (see sidebar WARNING). Either way, you should select and/or create any users that you wish to inherit the properties of this Site Group, and then click on the **Next >** button.

That's it for setting up the Site Group itself. Go ahead, and click on the **Finish** button.

Now, let's make sure that we have an Undocked User License to work with. Go to:

Tools
Configure ►
License Manager...

Now, go ahead, create an Undocked User License if you don't already have one, and then close the License Manager.

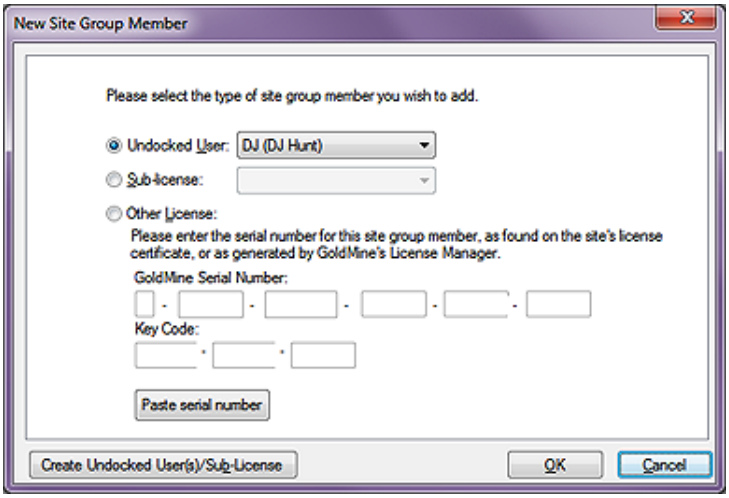


Figure 14-17

Next we'll move back to the GoldSync Administrator as I want to show you one of the overrides that may be beneficial to your organization. I would have you right-click on the Site Group that you previously created, and select **New...** ► from the local menu, and then again select **New Site**.

This action will immediately bring you into another Wizard, that of the **New Site Group Member** Wizard, Figure 14-17, and the alphabetically first of your UserIDs is presented to you for consideration in **Undocked User:**.

Figure 14-18

You may click on the **OK** button to just proceed here, and the Wizard should bring you to the **Welcome to the GoldSync Site Wizard!** dialog form page as shown here in Figure 14-18. You will notice that this dialog form supplies a bit more of the information about this member:

Site Type: Undocked User
Site ID: PKT55-PJ686-1RQ6L - DJ

You may change the descriptive name for any of the individual sites in the **To begin, please enter a descriptive name for this site.**. By default, on undocked users, the **&UserName**, and **&UserFullName** (both GoldMine Macros - see Appendix B) are selected as in my case resulting in **DJ (DJ Hunt)**. Additionally, you will notice that the option, I will ☑ **Allow this site to synchronize** is, by default, selected. Later, if I want to stop an individual member from syncing, I simply edit the properties for the member, and deselect this same option. Our goal today, however, is to set up this individual to synchronize.

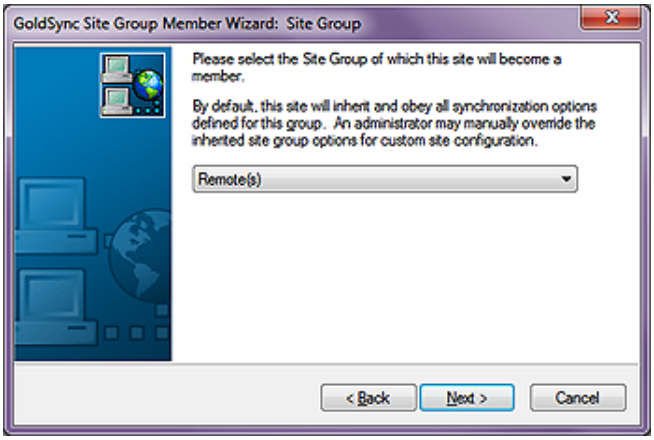


Figure 14-19

Clicking upon the **Next >** button will bring you to this **Site Group** dialog form page of the Wizard, Figure 14-19. This is redundant you say.

Tip

*Uncheck the ☑ **Allow this site to synchronize** option if you would like to create the site first, test it, and after you are satisfied that it is functioning properly, select this option to permit synchronization for the site.*

Well, yes it is in this case, however, this Wizard is also employed as a stand-a-lone Wizard for adding members to any of the site groups that you may have defined. It is, therefore, a necessary page for this Wizard.

I am simply going to click on the **Next >** button to advance to the next page of the Wizard, and I do hope that you will do the same.

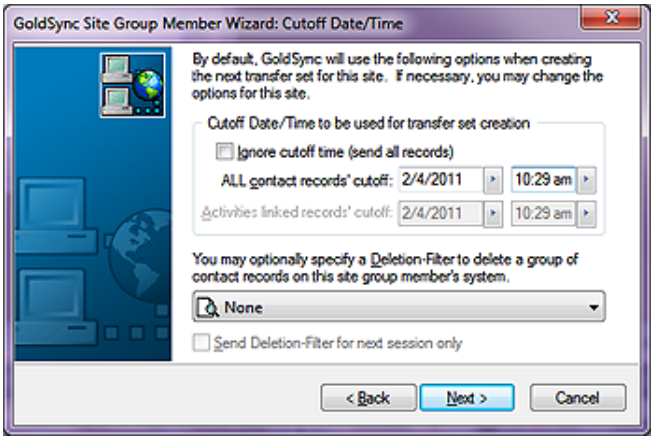


Figure 14-20

This action brings you to the **Cutoff Date/Time** Wizard dialog form, as shown here in Figure 14-20. You will notice that there are two distinct areas. The top most is the **Cutoff Date/Time to be used for transfer set creation**. This frame allows you to set the initial cutoff date, and cutoff time to be utilized. After each successful sync with GoldSync, the GoldSync application will update this date, and time automatically to the date, and time of the last successful sync.

Well, our first option in this frame is to ☐ **Ignore cutoff time (send all records)**. In past versions of GoldMine, selecting this option has been troublesome, and I advised my clients not to select this option for that very reason. I know that if I would like to get all records to a new site on the first synchronization that this would seem the obvious choice for doing that, yet I prefer the tried & true methodology that I have always employed.

Let's look at another way of doing the very same thing. In Figure 14-20, the current setting for **ALL contact records' cutoff:** is **1/28/2011 11:21 am**. This area allows one to change the specific cutoff date and time, so if one were to set this date back to a time before GoldMine existed, one should, and does, capture all records into their GoldMine transfer set. As VAR's, we are in the habit of using **1/1/1980** (**BG** - **B**efore **G**oldMine) as a good initial cutoff date. GoldMine was first released in 1988, and this date is certainly prior to 1988 or any date that you could possibly have created any of your GoldMine records without having previous modified your System Date.

The next section is unlabeled, but is the option to send a deletion filter to a Remote for deleting records on the Remote-side without having those deletions removed from the Server-side location on the Remote-sides next synchronization effort. One might use this if a group of records were being removed from one Remote-side location, and being reassigned to another Remote-side location. In the drop down list, one may select any predefined filter from any user. Should you select this option, the next option will be enabled. You would probably want to select the ☐ **Send Deletion-Filter for next session only** option, however, if you want the Deletion Filter to process on each pass then you would probably not want to select this feature. If the remote user has left the company, then you would want to allow them one last sync to remove all records from their GoldMine. Naturally, this would not remove any backups that they may have created. You would then remove them from the Site Groups list, preventing them from syncing with your GoldMine ever again. However, in case you haven't removed their site from the list after their last synchronization, you would not want to have this option selected. You would want to send the deletion filter until such time as you could remove them from the site group. On the other hand, if this person will be remaining with your organization, and you are just removing a selected few records, then, yes, you would probably want to select this option.

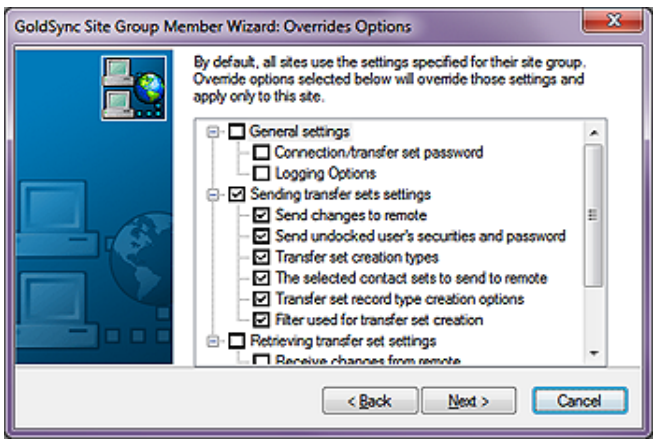


Figure 14-21

Clicking on the **Next >** button moves you to the dialog screen shown in here as Figure 14-21. From here, one is allowed to override every single option that has been set for the Site Group, however, it will be for an individual as opposed to the Group.

Tip

*David Lee says: The Territory Realignment Wizard in GoldMine will automatically create a deletion filter so, if you use the Wizard, you may not need to check the ☐ **Send Deletion-Filter for next session only** option.*

The big difference between using the deletion filter rather than simply deleting the records on the Remote-side is that you can easily reestablish the records on that Remote-side if needed. If you delete them, a transaction log is created that prevents you from reestablishing the records, and also create a deletion transaction that could delete those records Server-side, depending on the Server-side Retrieval settings.

For this entire Group, I will utilize the Group Retrieval options. I am only after one thing with this override, and that is the ability to vary what the Server-side will send to each Remote-side. To do this I simply select the ☐ **Sending transfer sets settings** when then, in turn, selects all of the sub branches as shown in Figure 14-21 on the previous page. Nothing else is required for this particular override session, hence, I would ask you to click on the **Next >** button again to advance to the next dialog form shown here in Figure 14-22 as the **General Options** dialog form.

As you chose to override the Send options the ☑ **Send changed data to this site** option is automatically selected for you. Please note that the ☑ **Retrieve changed data from this site** is also selected yet it does appear to be disabled. In fact, it is not disabled, but as we have not selected to override the retrieve options GoldMine will default this remote to using the retrieval options established for the Site Group. The same holds true for the **Logging Options** frame of this dialog form.

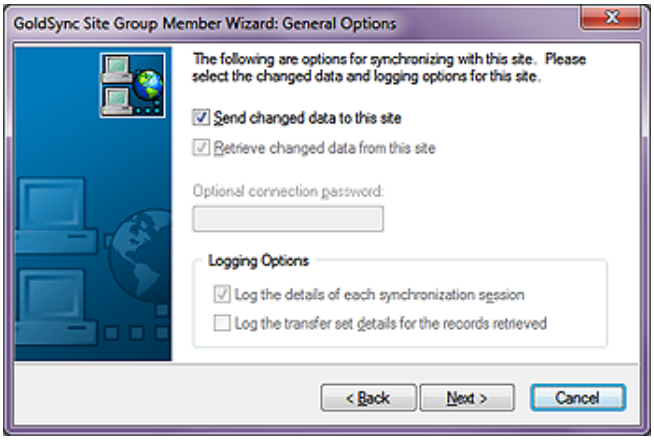


Figure 14-22

Click on the **Next >** button, and move on to the **Send Options** dialog form, Figure 14-23. Recognize this form? You should as it is the same dialog form that we discussed using Figure 14-11 earlier. On the override you may now select just those options that you wish to send to this remote only. Remember that you are overriding the Site Group Send Options for the individual user. In this case I have selected everything save the Logs. As there is no trouble to analyze as yet, there is no need to send the Server-side GoldMine Transaction Logs. For that matter, I never know of any reason to send the Server-side GoldMine Transaction Logs to the Remote-side. In cases of trouble, one might want to send the Remote-side GoldMine Transaction Logs to the Server-side for analysis, but I have yet to find a need for the reverse of that. Whereas, I always, yes always send **(all)** users' calendars.

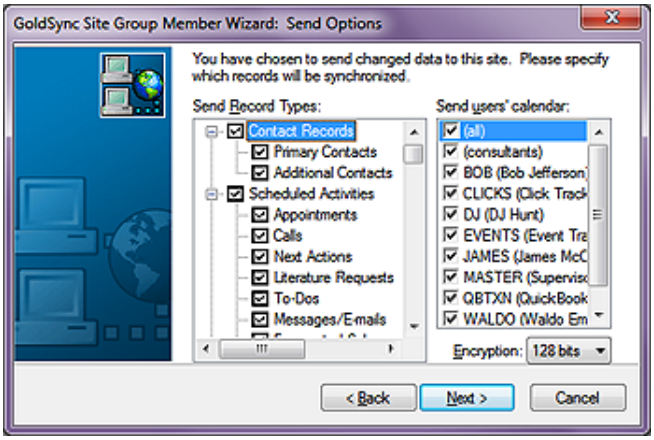


Figure 14-23

Click on the **Next >** button, and move on to the **Send Contact-related Options** dialog form, Figure 14-24. And again this is the same dialog form as that which we discussed using Figure 14-12 earlier. In this case, we will just accept the defaults unless, against my advise, you had a separate Contact Dataset Set for each UserID at which point you would wish to select the appropriate dataset. We will handle the proper data to be synchronized on the next dialog form which is the **Send Filter Option**.

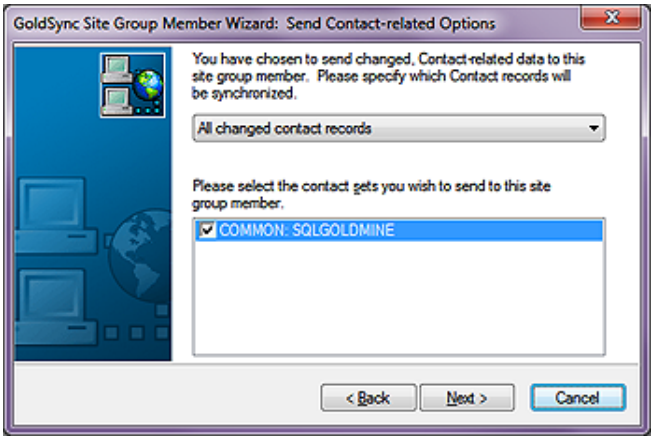


Figure 14-24

Click on the **Next >** button, and move on to the **Send Filter Options** dialog form, Figure 14-25 on the next page. And again, this is the same dialog form as that which we discussed using Figure 14-13

Tip

Name your Filters more meaningfully. For Instance: I probably should have named the Filter shown in Figure 14-25 something on the order of:

PDA Sync Filter for DJ

Figure 14-25

earlier, however, different from the Site Group configuration, we will utilize a filter for this Remote-side override.

As this was an override for my personal UserID, and as I already had a synchronization filter established for myself, I simply selected that filter, **PDA Sync Filter**, from the list.

Click on the **Next >** button, and move on to the final dialog form which I have chosen not to depict here. What can I say about this dialog form? There is nothing for you to do except click on the **Finish** button. Please do so if you have been following these procedures.

Note

I feel that the GoldMine One Button synchronization is too restrictive, to iffy and, therefore, I have no plans to cover its use in this book.

For Instance:

The size of the One-Button synchronization package may not currently exceed 2 Gigabytes.

At this point you may or may not be asked: **You have configured an 'Undocked' site member, would you like to configure a 'One-button Sync Profile' for this remote site?**

One-button Sync Profile, hmmmmmmm! This might be the way to go on an initial installation, however, I have never really employed this feature nor advocated its use. A lot of resellers swear by it, and as many that love it there will be hating it. For this exercise, I am going to ask you to select the **No** button as I had said previously, I set my cutoff date back to 1/1/1980 in order to get all of the required records to my Remote-side users. In fact, in the next section I will explain how I might initially set up a Remote-side. As you may expect, the initial synchronization for each Remote-side location may take an extremely long time, however, subsequent synchronizations, if done daily, will only Send/Receive the deltas, hence, they should take only a minute or two when using the IP to IP synchronization method.

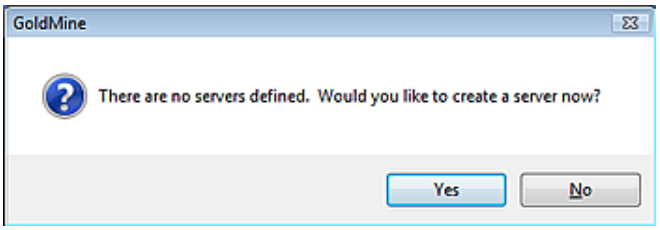


Figure 14-26

If you received that dialog form then clicking on the **No** button may bring up this next informational dialog form. If you are on the system that will serve as your GoldSync server, then I encourage you to select the **Yes** button here, otherwise select the **No** button. If you have selected the **No** button, every time that you start the GoldSync Administration Center, this informational dialog form will pop up anew. I am using my GoldMine server as my GoldSync server as well, therefore, I am going to click on the **Yes** button. This action will start a new GoldSync Server Wizard from which I will configure my GoldSync Server, and you, your GoldSync Server.

Tip

Setting up the GoldSync Server is much more efficient if done while sitting at the GoldSync Server.

Note

Please note that it is not imperative that the GoldSync Server be running any Windows Server operating system. It could just as easily be, and usually is, an idle Workstation on the Network that has a fixed IP Address.

Figure 14-27

Notice here in Figure 14-27, that the first page of the Wizard has automatically picked up the name of my server, which, as this is an example, happens to be my Workstation upon which I am typing this book (refer to sidebar Tip). This is why I asked you to be on the system that will be your GoldSync Server, although, you could have, as easily, just typed in the name of the server if it were different than the system on which you were working. I haven't had any issues running it this way, hence, this is the method by which I am choosing to instruct you. Click on the **Next >** button to advance to the next page of the Wizard as shown in Figure 14-28 on the next page.

It is from this dialog form, that one may set up any or all of five different types of processes for GoldSync. The most common one, and the one that I have selected is ☑ **IP to IP/Network: Accept incoming remote connections**. This allows for the synchronization of GoldMine over the Internet or over the Network to the IP address of the GoldSync Server. This option causes GoldSync to sit there, and monitor, once started, the IP Address waiting for an incoming connection request from another, outside, IP address. As I said, this is, by far, the most common configuration for the GoldSync Server.

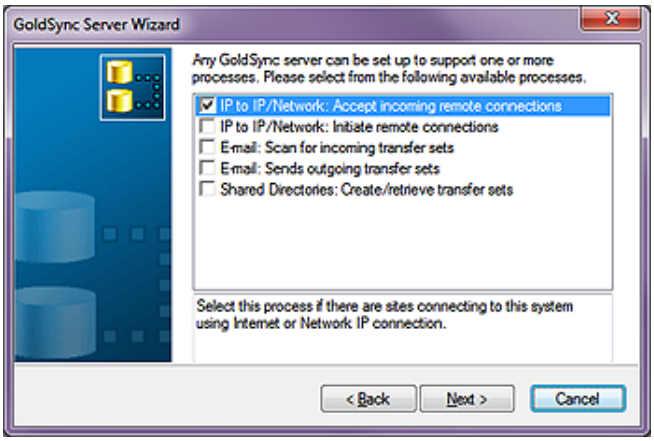


Figure 14-28

The descriptors are relatively self-explanatory, however, I would like to discuss, in a little more detail, the 5th option, ☐ **Shared Directories: Create/Retrieve transfer sets**. This option will create, in advance of the actual synchronization by the Remote-side, a transfer set for the Remote-sides in shared directories. If you have users who sync infrequently, you may want to employ this option. If this option is employed, the user will not need to stay connected to the sync server, and wait for the sync server to build them a transfer set. They will just need to pick up a waiting transfer set after they have dropped theirs off. This option also works well when you have slow connectivity such as dial-up (does anyone still use dial-up?) Internet access.

Tip

To keep synchronization time to a minimum, I highly recommend that your Remote-side users synchronize at least once each work day, if not more frequently.

At this point, I would ask you to only select the first option for now, and then click upon the **Next >** button which will bring one to a **GoldSync Server Wizard** dialog form where one can do little more than select **< Back** or **Finish**. Naturally, I would select **Finish** at this point.

That was rather painless now, wasn't it? You're already finished, or are you? Clicking on the **Finish** button on the last dialog form of the current Wizard will begin, yet another, Wizard. In this case the **GoldSync Process Wizard**, which, as you've already surmised, is display here in Figure 14-29.

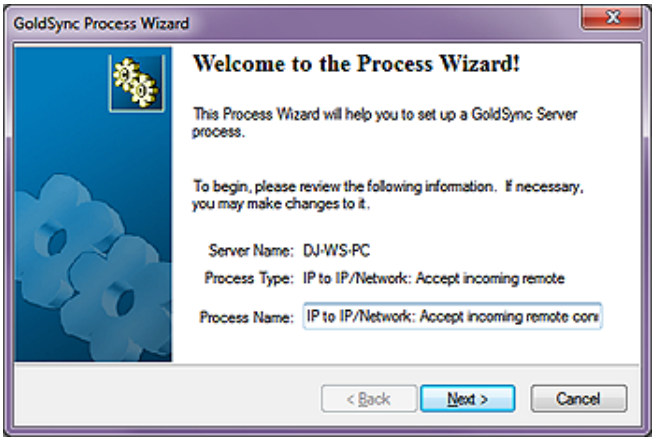


Figure 14-29

You may change the **Process Name:** descriptor on this page, however, I usually just accept the default name, and click upon the **Next >** button. Doing so, will advance you to the dialog page shown in Figure 14-30.

Tip

David Lee says: Synchronization is very resource intensive. You should be sure that you have about 0.5 GB of RAM for every concurrent process. Thus, if you have 2 GB of RAM on your server, you should set the maximum number of concurrent processes at 4 for optimum throughput. Running too many concurrent processes with too little RAM could lock the processes. GoldSync does not give you a warning, it just stops working , and you will need to restart in order to recover after you have realized that you have the problem. This is based on testing by resellers in live environments, and not on FrontRange Systems recommendations.

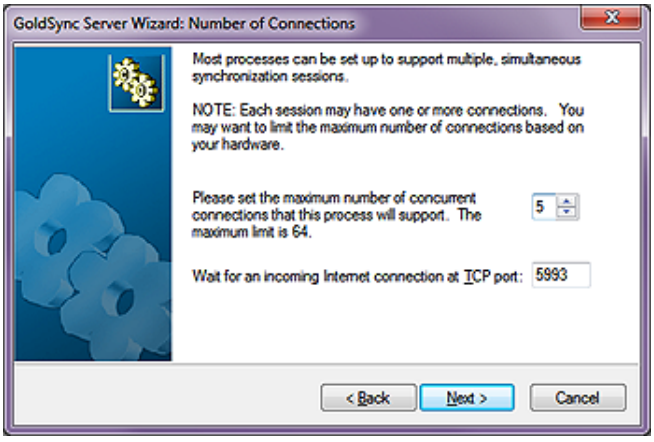


Figure 14-30

There is little that one may do on this, Figure 14-30, dialog form page as well. Your first option is to set the maximum number of concurrent connections that this process will support (refer to sidebar Tip). The default is 5 connections, while you could have as many as 64 connections handled by this one server, the server would need to be a dedicated and powerful server to handle that many concurrent connections. I believe

that FrontRange recommends no more the 10 concurrent connections per server as the performance optimum.

Whatever you want to set, let's just say that, for now, we accept the default of 5 concurrent connections, this is the maximum that can connect consecutively. If a 6th user attempts to connect, while there are 5 others connected, the 6th connection will be refused by the GoldSync Server. They will need to try again at a later time. This is a way to help prevent your server from running out of resources. Many people believe that if they have 10 people synchronizing, that they need to create 10 concurrent connections. This is false. You need to determine how many users are likely to synchronize at exactly the same time. Concurrent is definitely the key word here.

Note

The combination of the IP address and the Port address is referred to as a socket. The socket might be:

192.156.254.27:5993

Unless you have an excellent reason to change the **TCP port:** from the default **5993**, I suggest that you do not. This is not a common port to begin with. Regardless of your selection, GoldSync will monitor the port set on the IP address for all connection requests.

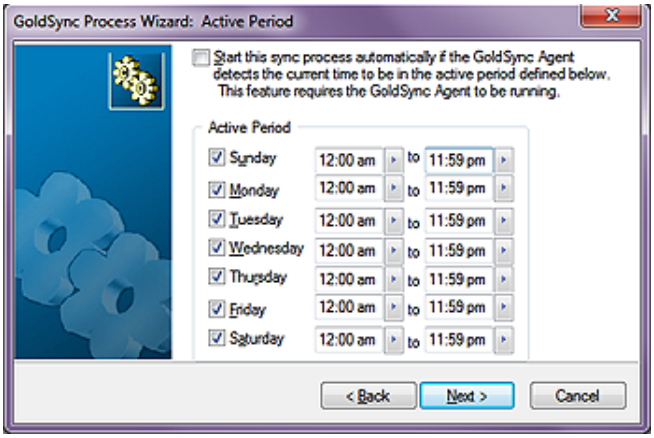


Figure 14-31

Clicking on the **Next >** button will bring you to the **Active Period** dialog form as is shown here in Figure 14-31. The ☐ **Start this sync process automatically...** is normally not selected by default, and it is my opinion that it probably should be selected by default. I always recommend selecting this option if you are establishing only a single GoldSync Server, after all, we do want GoldSync to be as automated as is possible. This means that, should anyone start GoldSync in the **Silent Mode** or as a **Service**, that this process would be automatically started. Also, if you highlight this process in the **GoldSync Administration Center** when you are through, you will notice in the toolbar that there is an enabled **Start** button, and a disabled **Stop** button. You could click the **Start** button to this GoldSync Server immediately even while running GoldMine. This would be in lieu of running GoldSync as a separate instance in the **Silent** or **Service Mode**.

Note

The commandline for starting GoldSync via the Windows Task Scheduler is:

G:\GoldMine\GMW.exe /u:UserID /p:Password /s:GoldSync

Remember to leave a single space before each forward slash (/).

The **Active Period** frame is used to define when GoldSync will be active, and when it will not be active. To be honest, I leave this alone, and accept the defaults as I want the Remote-side users to have the ability to synchronize whenever they are connected to the Internet. I start GoldSync via the **Windows Task Scheduler**, and as well, I stop it via the **Windows Task Scheduler** using an application known as **Close_GM.exe**. However, if you are using the **GoldSync Service** feature, which I do not recommend at all, then I highly recommend that you do define specific active periods.

Note

*Need a copy of **Close_GM.exe**?*

Send an e-mail to:

DJ@DJHunt.US

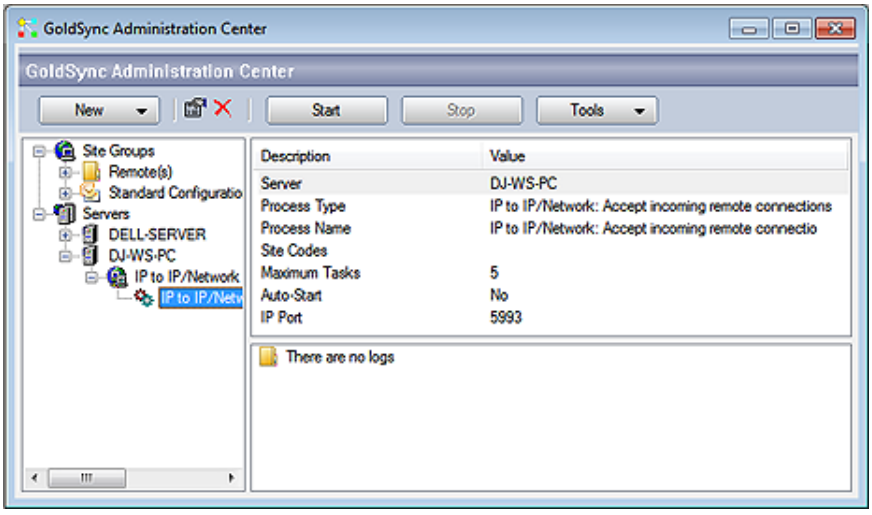


Figure 14-32

Clicking on the **Next >** button in Figure 14-32 will advance us to the **Finish** dialog form, and we are at that point again. Does **Finish** actually mean finish this time? Yes it does. Please click on the **Finish** button to complete this process once, and for all.

Initial Remote-side TSets

At this point, you should be returned to **GoldSync Administration Center**, and it should appear something like that shown on the previous page in Figure 14-32. From here, you may **Clone...** additional sites, create a **New Site Group**, create a **New Site**, create a **New Server**, create a **New Process**, or you may **Create Installation File for One-button Sync...**, which, as you remember, I passed on before. There are many other activities that you may work on within the GoldSync Administration Center, but we do need to move on.

At this point in my writing, I am going to deviate from my previous book outlines to add this new section called **Initial Remote-side TSets**, and this is to answer that question: *"Well, if you don't recommend One-Button Synchronization then just how do you set up a new Remote-side user?" So here you go.*

Step 1:

☐ I create an Undocked User License for the Remote-side, and I copy that to a Notepad document.

☐ I create their Sync Site from the GoldSync Server or a Workstation, and don't forget to include any Synchronization Filter if you are not sending the entire GoldMine dataset.

☐ From the GoldSync Server or a Workstation I create TSet of just the Customizations in the Server GoldMine:

Tools
Synchronize ►
Synchronization Wizard...

Figure 14-33a

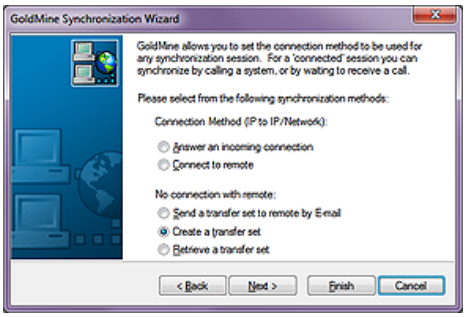


Figure 14-33b

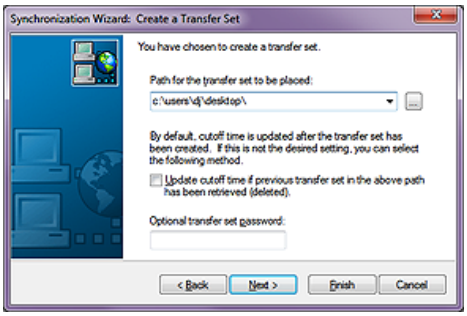


Figure 14-33c

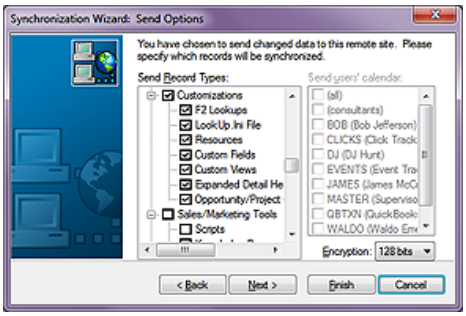


Figure 14-33d

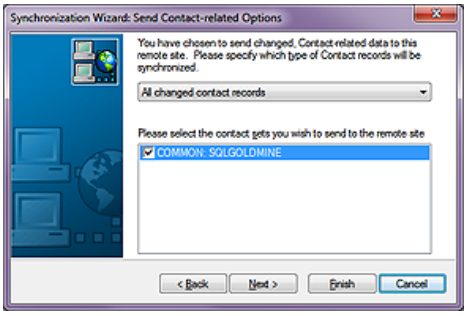


Figure 14-33e

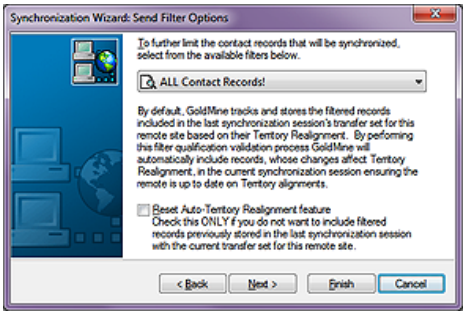


Figure 14-33f

You may not have noticed, however, in Figure 14-33d I first right-clicked over the **Send users' calendar:**, and selected **Clear All**. I then right-clicked over the **Send Record Types:** tree, and selected to **Clear All** there as well. Lastly, I scrolled down to the ☐ **Customizations** group, and simply checked that option. That is all that was selected in Figure 14-33d even though you can't see the entire Tree in the image.

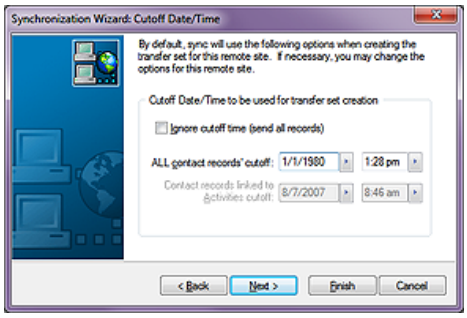


Figure 14-33g

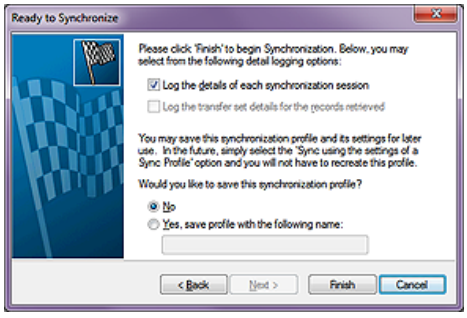


Figure 14-33h

You may have noticed, Figure 14-33g, that I changed the **All contact records' cuttoff:** to **1/1/1980**. This is a date before the existence of GoldMine on this earth.

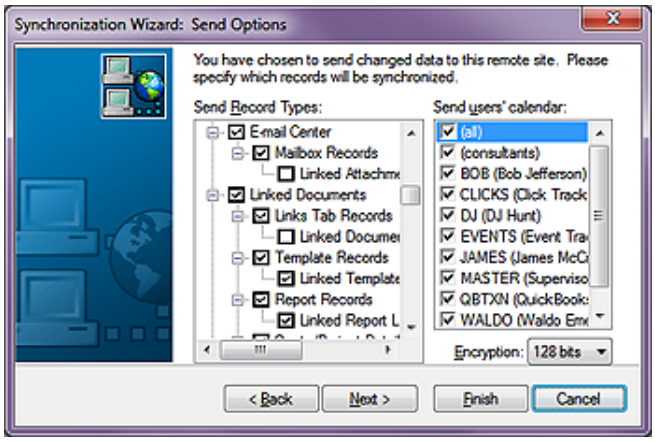


Figure 14-34

☐ From the GoldSync Server or a Workstation I create TSet one more time only this time I want to forgo the Customizations and any Linked Attachments, see Figure 14-34. If this Remote-side is not to get the entire GoldMine database then now would be a good time to implement a Filtered Set for this Remote-side Transfer Set. Also, make certain that the cutoff date is set to **1/1/1980**.

☐ Next, I would zip up the Folders of Linked Documents (there's no sense in trying to include those in your TSet).

Let's create the bundle now for this Remote-side. In a downloadable location include:

- ☐ **Notepad document containing Undocked License Number**
- ☐ **Customizations Transfer Set**
- ☐ **All/Filtered Records Transfer Set**
- ☐ **Zipped file containing Linked Documents**
- ☐ **GMSetup.exe equivalent to Server-side GoldMine Build**
- ☐ **Appropriate SQL Server 2008 Setup files**

Server-side, I think that would be everything. Let's move to the Remote-side for this Site Group.

☐ Download all of the files to the Remote-side Desktop from the ftp site.

☐ Let's begin by installing SQL Server 2008. Make sure that you install SQL Server 2008 using the **Mixed Mode** authentication when you arrived at that dialog form in the Wizard. Personally, although I hear that it is not a good practice, I assign the **sa** login the same password as I have utilized on the Server as there is less chance of their forgetting it. However, Computerese Inc always assigns a back door login with Administrative Rights immediately after the installation process has completed for SQL Server. We always assign the **Login: GMUser** with our own common password. We do this knowing fully well that the client will eventually forget their sa login password.

☐ In the next step what we want to accomplish is to install the latest build of GoldMine Premium. Refer to sidebar WARNING before proceeding. It is essential that you maintain the structural integrity of the path to preserve your capability to successfully synchronize documents to and fro the Sever-side and the Remote-side. At the very beginning of this chapter I talked about the GoldSync.ini, and this is a critical action to facilitate document synchronization as employed via the GoldSync.ini.

☐ This may need to be part of the last step, however, open the Notepad document containing the Undocked or Site License, and **Copy** the license number to the **Clipboard**. If GoldMine did not start automatically, it probably did, then start it now. The first thing that you should be asked for is the license number. Simply click the button to **Paste License** (I'm not positive of exactly what the button caption is without doing a new installation this moment) number.

WARNING

There is one and only one structural location for the install of GoldMine on the Remote-side. Structurally, the resulting path must be:

<Drive Letter>:\GoldMine

So I really don't care if the resulting path is:

C:\GoldMine
D:\GoldMine
G:\GoldMine

as long as you maintain that structural paradigm.

☐ Before we do anything else, and while we are in GoldMine, we would like to establish out GoldSync.ini settings. These should be exactly as described at the beginning of the chapter. From the GoldMine menu:

Tools
Synchronize ►
Synchronization Settings...

You know what, it is such a critical step that I am going to Copy & Paste that paragraph here so that you don't have to flip back and forth:

> If you followed my tip, then the next option is where one would establish a rule that will compensate for the drive letter differences in between the Server-side path and the Remote-side path. For Instance; your Remote-side might save a linked document in: **C:\GoldMine\Documents**, while the comparable path on the Server-side might be: **Y:\GoldMine\Documents**. Since the file's original path, including drive letter, does not exist on the server, it is important that one suggests other possible drive letters. As I never recommend using UNC paths, \\Server\C\, I do not suggest that you incorporate any of these in this rule. All of the optional drives to be searched should be separated with a semicolon (**;**). Based upon my For Instance, I would want to include **Y** in this space on the Server-side configuration, while I would have used a **C** on the Remote-side configuration.

☐ Okay, we're moving forward. Next, using the Synchronization Wizard:

Tools
Synchronize ►
Synchronization Wizard...

We want to retrieve that **Customization Transfer Set**. I won't supply you the steps for that as it is a Wizard, and very straight forward.

☐ Can you imagine what our next step would be? Of course you can. Using the Synchronization Wizard, I want you to now retrieve the **All/Filtered Records Transfer Set**.

☐ Lastly, for these steps at least, I want you to unzip the file containing the Linked Documents, and unzip them to the exact same path from whence they came off of the server except that the drive letter should have changed.

And with that, your Remote-side should now be established properly. You have one more task to accomplish, which is to set up and do an actual Synchronization test to/from the Server-side. Before I can proceed with that section, however, I must talk about the IP Address.

Note

*It is very conceivable that you may have utilized a different drive letter for your Remote-side installation. Were I have entered a **C**, you will wish to enter the drive letter under which you have the Remote-side GoldMine installed. Again, refer back to my WARNING on the previous page governing this matter.*

Note

As an administrator, you may want to do due diligence and establish a Maintenance Plan for the Remote-side to execute via the SQL Management Tools at this point. And training the Remote-side how to utilize or monitor the process depending on whether you establish a Manual or Automatic process.

IP Address

Note

*Please make a mental note that I have assigned port **192.168.1.130** permanently to the GoldSync Server so that I will not have to constantly keep reconfiguring my Router Port Mapping each time the computers reboot.*

Note

Whether inside or outside the network, the Remote-side users must have full rights to the GoldSync Server as they will be Reading/Writing/Deleting from this server via the GoldSync application.

Note

*The combination of the IP address and the Port address is referred to as a **Socket**.*

The socket might be:

192.156.254.27:5993

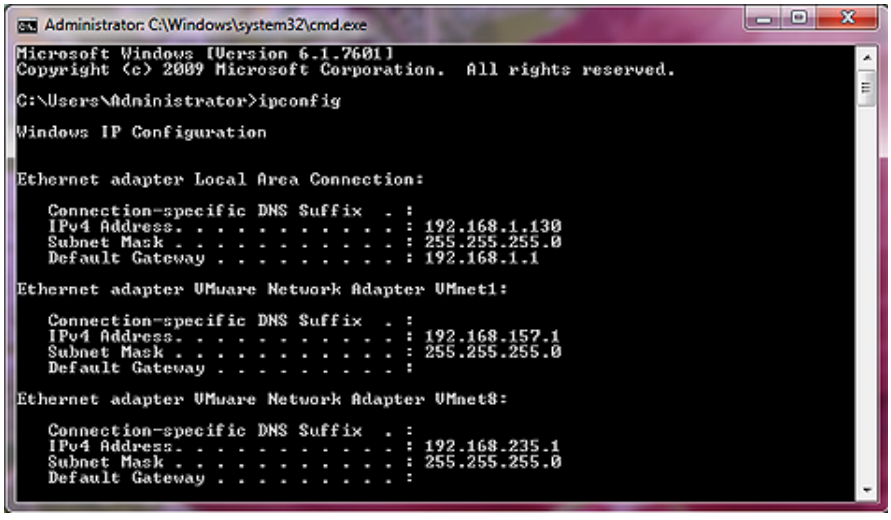


Figure 14-35

I want to comment on the **IP Address** that GoldSync is monitoring on the Server. When the Remote-sides synchronize to GoldSync using this process, they will be designating the IP Address, and the Port (Socket), to which they intend to synchronize. Finding the IP Address of the GoldSync server is easy. Go to **Start** | **Run...**, enter **Cmd** (older Windows versions use Command), then click the **OK** button. This will bring you into the DOS command window with a blinking cursor. If you enter **IPConfig** from your keyboard, and hit the **Enter** key, the display, similar to that shown in Figure 14-35, should result.

This depicts the internal IP Address for this system on this network. The first IP address, **192.168.1.130**, is of significance to us, and is the IP address of my GoldSync Server. Of course, I am using the default Port of **5993** when synchronizing my Remote-sides. This, then, is the IP Address that the Remote-side would use in their configuration when, and only when, they are con-

nected to the network either in-house or via a **Virtual Private Network** (**VPN**). This is the socket that GoldSync will be monitoring for connection requests.

When the Remote-sides are outside of my network, they must synchronize to the firewall IP Address on Port 5993, and the firewall must handle the redirection (**Port Mapping**) to the GoldSync Server IP Address which, as we already know, is 192.168.1.130. Because of the vast number of firewalls, and software, it is beyond the scope of this book to attempt to help you configure your firewall. Your Network Administrator should be familiar with this process, and should be able to assist you. Your Network Administrator must also assure that the Remote-side users have full rights on the GoldSync Server.

We have now positioned ourself in this process to establish a **Profile**, and synchronize any **Remote-side** user that has been established on the **Server-side** as a **Site**. For this section, I will assume that this Remote-side is connected to your internal network, and that they will be synchronizing across this network to a previously designated internal IP address. Additionally, I will be working exclusively on the Remote-sides system at this point. I will assume that the GoldSync Server is up and waiting for a remote connection request.

Obviously, I would want to make certain that the Remote-side can see the Server-side. Therefore, after establishing my network connectivity, I would ask you to bring up the DOS Command window on the Remote-side as we had done previously on the Server-side. This time, however, at the command prompt, I would like you to **Ping** the IP address of the GoldSync Server. That's right, an old Navy term to see if a response is returned by the pingee.

At the command prompt, I would like you to type (IP address of your GoldSync Server):

Ping 192.168.1.130

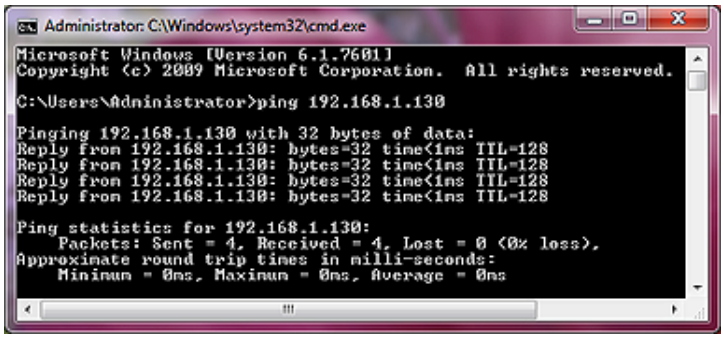


Figure 14-36

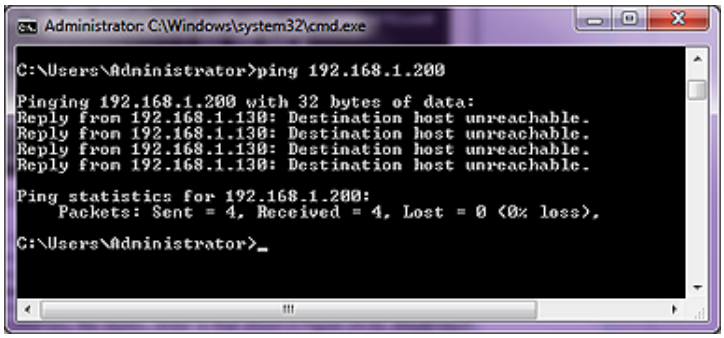


Figure 14-37

You would expect your results to resemble those shown here in Figure 14-36. If your screen looks similar to that shown here in Figure 14-37, then you do have a communication issue. Notice the **Destination host unreachable** at the end of each Reply statement, whereas, Figure 14-36 shows that we have received a Reply from 192.168.1.130, our pinged site. If you can't achieve the state shown in Figure 14-36 while using this Remote-side, then this Remote-side will never, ever be able to synchronize with your GoldSync Server.

This is a critical point that a lot of new installations tend to over look. If you can't see the GoldSync Server, then the GoldSync Server can't talk to you. The Remote-side may receive a **10060** or a **10061** error message, communication failure if this is the case. Often the Remote-side will believe that they have been synchronizing successfully, because they see the flurry of activity prior to their unit attempting to communicate with the GoldSync Server. Often the error message goes unnoticed by the Remote-side user, and they adamantly claim that they have synchronized their data. If so, why isn't it on the Server-side? Hence, we have the GoldSync Logs. These logs will give the GoldMine Administrator a way of ascertaining whether the user is telling the truth, or the user is, well, how do I put this, a normal end user.

To continue, I will assume that you have already done the initial setup of the Remote-side users system as described earlier. Pretty much now, all that is left for us to do is to set up the Profile for the Remote-side, and perform an internal test synchronization.

Note

My current Firewall Socket for GoldSync is:

74.104.173.223:5993

However, that is a moving target as it is a dynamic IP Address. To stabilize the IP Address, at no cost I might add, I utilize a service from ***www.No-IP.com*** *called* ***No-IP Free****.*

GoldSync Remote-side

WARNING

If you cannot Ping the GoldSync server from the Remote-side, then the Remote-side will never be able to synchronize with the Server-side.

Period. The end.

You can begin setting up the Remote-side synchronization process by proceeding to:

Tools
Synchronize ►
Synchronization Wizard...

This is found on the GoldMine menu on the Remote-side. This will begin the Wizard as shown here in Figure 14-38.

Figure 14-38

I will use the ⦿ **Start a new session** option which is the default when on a new GoldMine installation, and then I would ask you to click on the **Next >** button to advance to this page of the GoldMine Synchronization Wizard as shown in Figure 14-39.

Figure 14-39

As this is the Remote-side, I will select to ⦿ **Connect to remote**. You say "Huh? The Remote-side shouldn't be connect to the remote?". It is just a matter of semantics. Yes, the Remote-side wishes to connect to the remote GoldSync Server, which is, in fact, remote to this system (see sidebar Tip).

It should be obvious that you need to use one of the **Connection Method (IP to IP/Network):** options, just as it should be obvious that you cannot have both the Server-side, and the Remote-side waiting to ○ **Answer an incoming connection**. One has to be waiting for an incoming connection request, Server-side, while the other has to be requesting the connection, Remote-side.

Click on the **Next >** button to advance the Wizard.

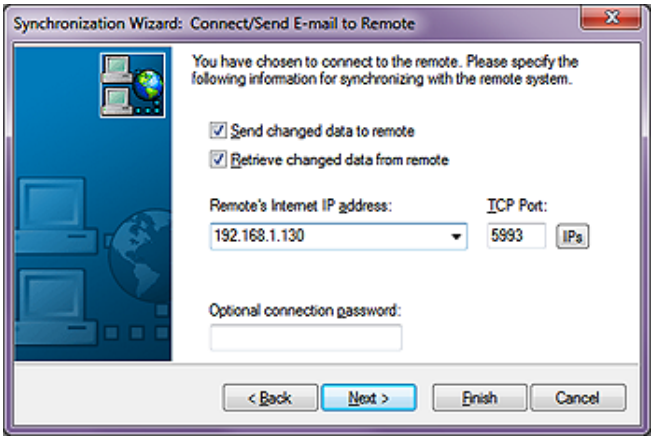


Figure 14-40

This brings us to the **Connect/Send E-mail to Remote** page of the Wizard, Figure 14-40. Forget about the semantics. I would ask you to select, if they are not already selected, ☑ **Send changed data to remote**, and ☑ **Retrieve changed data from remote**. Remember it is the intention of this chapter to synchronize all changes to/from the Server-side.

I would also ask you to remember that I am connected to the internal network for this exercise, and, previously, I asked you to find out, and test the internal IP address of the GoldSync Server. I ask you now to enter the IP address, successfully tested in Figure 14-36, into the **Remote's Internet IP address:** field. You could just as easily entered **DJHunt.No-IP.biz** which is not an internal IP address, but my No-IP static IP Address for connecting when one is on the road. I just want to show you that if you don't have the money for a fixed Router

Note

*Remember what I said about remembering your seating position. You are setting up a Remote-side, so even though the GoldMine Wizard option is saying ⦿ **Connect to remote**, the remote that GoldMine is eluding to, in this case, is the Server-side as it is remote to this unit.*

*Remember where you are sitting when you are running the **GoldMine Synchronization Wizard**.*

IP address, as I do not, that there are free ways around this problem. Go to www.No-IP.com, and pick up a free No-IP redirection account which will emulate having a fixed IP Address. Oh, and don't forget to download DUC.exe as this is what broadcasts your dynamic IP Address back to No-IP for the emulation process.

As long as you made no changes to the **TCP Port:** on the Server-side, I would ask you to accept the default port of **5993**.

As I did not add a connection password while setting up the Server-side, I will not add the **Optional connection password:** on the Remote-side. Personally, the transfer set is encrypted, and having a password seems redundant, although, when employees leave the company, I suppose this could be a good thing to prevent them from continually synchronizing. Then again, that would be redundant with the use of a Deletion Filter. I don't know. Six of one, and a half dozen of the other. You decide what is the best for your situation.

WARNING

It is critical to the proper GoldSync operation that the TCP Port Server-side and Remote-side match exactly or you will receive a 10060 or 10061 connectivity error.

Tip

*Even though I have asked you to **Select All**, you may want to consider turning off (unchecking) the **Logs**. These can grow abnormally large when not Purged Daily, and are not necessary to send from the Remote-side unless you are experiencing issues with synchronization that you wish to analyze.*

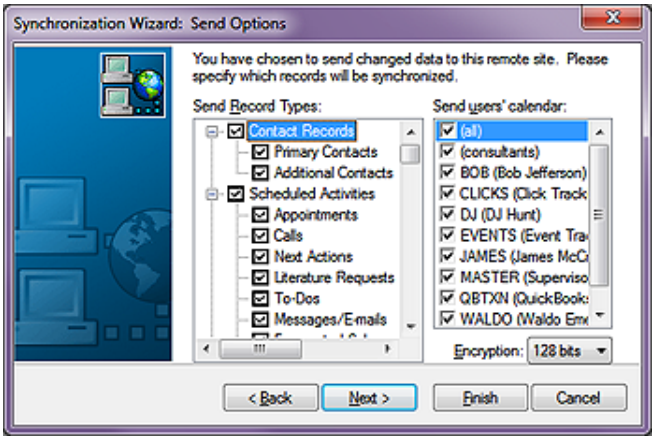


Figure 14-41

Click on the **Next >** button to advance the Wizard to the **Send Options** page, Figure 14-41. For this dialog form I, again, ask that you right-click on the **Send Record Types:** list, and choose **Select All** from the local menu. I also ask that you check the ☑ **(all)** box in the **Send users' calendar:** list.

If you did not change the **Encryption:** on the Server-side, then do not change it here, and you may click on the **Next >** button to advance to the **Send Contact-related Options** page of the Wizard shown in Figure 14-42.

Tip

*I only advocate the use of a single contact dataset except in rare cases, and those rare cases usually do not have to be synchronized out to the Remote-side. In Figure 14-42, you will notice that I have one dataset. It is named **COMMON: SQLGOLDMINE**.*

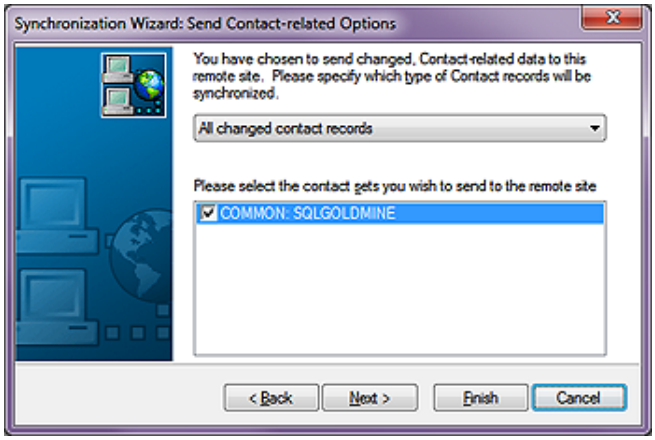


Figure 14-42

The first selection is a drop list containing four possible selections:

Only the current contact record

All changed contact records

Contact records linked to the 'Send user's calendar' list

All filtered records and user scheduled activities' records

As you can easily see, I selected the second item, All changed contact records.

Note

*As of GoldMine 6, all datasets must have a **File Code**, otherwise you will not be allowed to set up a synchronization profile. Each File Code must be unique. You may have noticed, Figure 14-42, that I used **COMMON** as my File Code. It doesn't matter what you use as long as each is unique and within the fields character limitation. Personally, I prefer something that is descriptive of the dataset. You should be aware that I have the identical configurations Server-side, and Remote-side.*

One is allowed to synchronize more than one dataset, and, as of GoldMine 6, you are required to use unique **File Codes** for each of your datasets created. For this exercise I will only be synchronizing the one dataset, hence I have checked my primary dataset ☑ **COMMON: SQLGOLDMINE**.

Note

***Send Filters** are more apt to be used under the Server-side **Send Filter Options** than they are here on the Remote-side.*

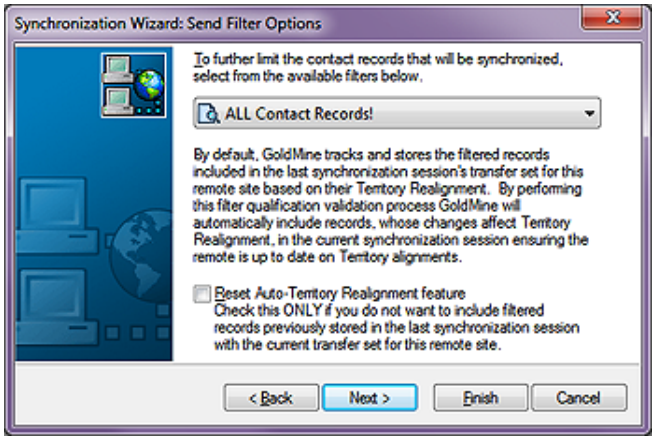


Figure 14-43

I then ask you to click on the **Next >** button to advance to the **Send Filter Options** page of the Wizard, Figure 14-43. This is the Remote-

side (always remember where you are sitting when you create your profiles), therefore, I would want them to send everything that they have done back to the Server-side. The Server-side can then pick, and choose from the transfer set that which it wishes to retrieve. You may choose, as I have, **All changed contact records**, or you may choose from any of the available filters.

There is no longer an Options button as you may remember from previous builds of GoldMine.

The last option on the Send Filter Options dialog form, Figure 14-43, is to check, or not, ☐ **Reset Auto-Territory Realignment feature**. On the Remote-side, I have never had the need to select this option.

I now click on the **Next >** button to proceed to the **Cutoff Date/Time** dialog form of the Wizard, Figure 14-44. As this is a new installation of GoldMine on the Remote-side, and this is my first synchronization with the Server-side, I really don't need to send anything from the Remote-side to the Server-side. You must constantly position yourself mentally at the unit which you are setting up. I am setting up the Remote-side Send options, the converse of which is the Server-side Retrieve options. I will, therefore, accept the defaults on this page, and advance to the next page of the Wizard by, again, clicking on the **Next >** button.

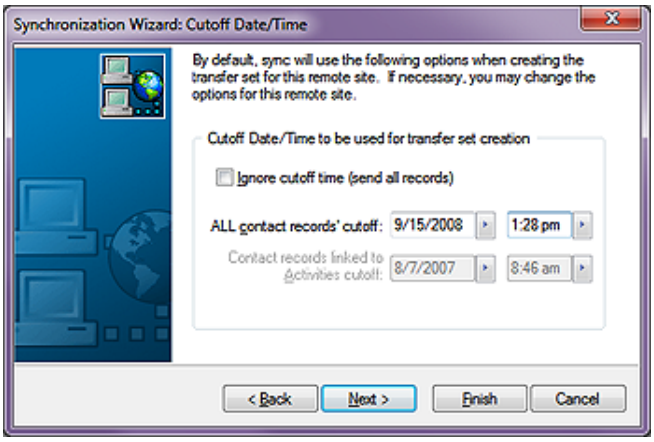


Figure 14-44

I am now ready to begin with the **Retrieve Options**, Figure 14-45. In most cases I would expect that the Remote-side will want to retrieve everything that the Server-side has sent to it. I simply right-click individually in both trees, and choose **Select All** from the local menu on both list. I will usually then uncheck any **Logs** on the **Retrieve record types:** side.

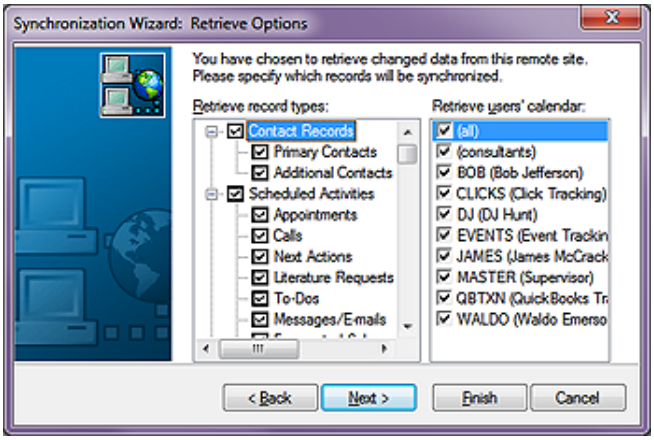


Figure 14-45

Clicking on the **Next >** button brings one to the **Retrieve Contact Sets** dialog form of the Wizard, Figure 14-46. You must have a contact set chosen under Default contact set, if the incoming contact set is not specified, and, as implied, you do not have to check a dataset. If, however, you are synchronizing more than one dataset, then you must select those datasets that you wish to retrieve from the Server-side. Simply said, select all of the datasets that you wish the Remote-side to accept (retrieve) from the Server-side, and then click on the **Next >** button.

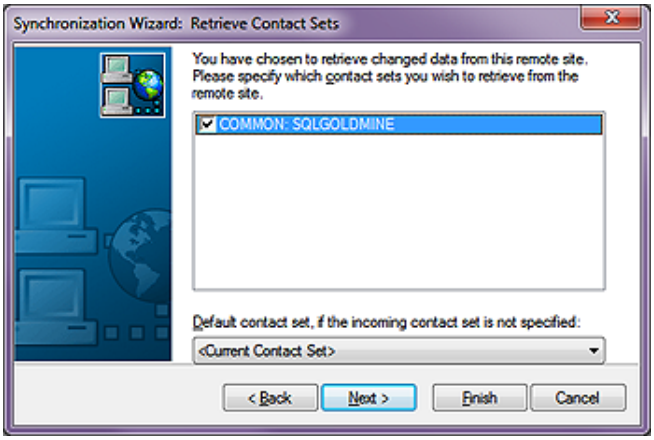


Figure 14-46

This brings us to the **Connect/Send E-mail to Remote** page of the Wizard, Figure 14-47 as shown on the next page. From this dialog screen I would select when to connect. The default is ⦿ **Immediately** which I have accepted.

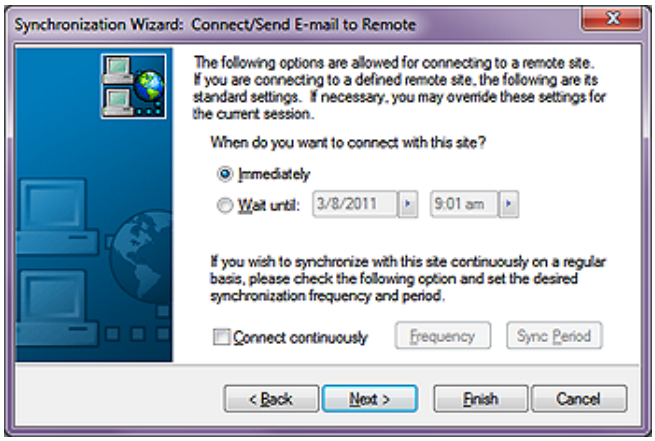


Figure 14-47

I would like to mention the ○ **Wait until:** option. Obviously, this will allow GoldMine to wait until the specified date and time couple prior to connecting to the Server-side if, and this is an important if, this instance of GoldMine is actually running on the specified date, and time couple. Personally, I have never had the opportunity to avail myself or any of my clients of this feature, but it is there should you have a need for such an option in your particular situation.

If your Remote-side users are connected to the network, even if they are not, you may want to select the option for them to ☐ **Connect continuously** which, in the default state, is not selected. As you can see, once selected, you may set a **Frequency**, as well as a **Sync Period**. The frequency can be set in days, hours, and/or minutes. The sync period is the setting of the hour ranges on Sunday thru Saturday when the Remote-side should be trying to connect to the Server-side. I don't ever recommend this option, and believe that it should be controlled Server-side. I would, therefore, not select this option at all. I would remain with the default connection method of immediately, and I would click on the **Next >** button to advance to the next dialog page of the Wizard.

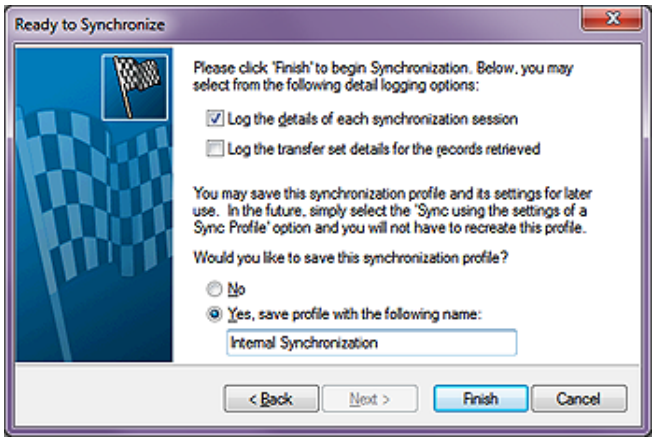


Figure 14-48

Yes, you have reached the final page of the Remote-side wizard, and you are now **Ready to Synchronize**, Figure 14-48, when you click on its **Finish** button.

As this is the first time that the Remote-side would be synchronizing with the Server-side, however, I would like to make some changes to this page. You will notice that I have selected to ☐ **Log the details of each synchronization session**. After I am sure that things are running as expected, I may wish to deselect this option. For the time being however, I would like to make sure that everything is running smoothly, and the logs will validate that for me. I can do this from Server-side by reviewing the logs on the Remote-side if they have been saved. For a while, I would leave this option selected.

As I am setting up the Remote-side (remember where you're sitting), I would also want to save this synchronization profile for their later usage. I chose to save this as **Internal Synchronization** as I am syncing while connected to the network inside the firewall. You may want to set up two of these profiles. If you now have the Remote-side connected to the internal network for this initial synchronization, then you may want to set up one profile to Internal Synchronization, and another to Remote Synchronization. The second would be for when they are syncing from the road (outside the firewall), and will need to use a different IP address.

Once this dialog form is completed, a simple click on the **Finish** button will save the profile, and then proceed to perform the first synchronization. However, make certain that the Server-side is **Listening for incoming IP connection (x connections...)** before you click on the **Finish** button on the Remote-side.

Watch the process monitor. It will process the Remote-side first, and then attempt to connect to the Server-side. This is where the error will most likely occur if it is going to. You may see a **10060** or a **10061** connectivity error if the Remote-side cannot see the server-side IP address, or the if the Server-side is not set to Listening for incoming IP connection (x connections...). This is the activity that few Remote-sides catch. They see the processing on their side working just fine, yet they fail to see the connection error and they say "Yes I've synced to the server just fine." when, in fact, they did not.

Note

Logs are continuously being appended to with new records. Contrary to FrontRanges recommendations, I recommend that you ***Purge Logs*** *after successfully synchronizing. The default setting is to save logs for one month, and I recommend that you accept that setting.*

Remember, logs should be purged Server-side and Remote-side on a regular basis.

Note

After a synchronization has been run once, and the profile saved, the next running of an unchanged profile is even easier.

All that you, or the user needs to do is:

Alt - T
Enter
S
S
Alt - P
Tab
Up/Down arrow to desired profile
Alt - F

These also represent the key strokes for creating the keyboard macro. So why not create one?

You should be advised that the addition of a new Synchronization Profile after the creation of the macro could break the macro.

After you have completed your synchronization with the Server-side from the Remote-side, I recommend that you create two keyboard macros to run each of the synchronization profiles. One button for synchronizing while on the network, and the other for synchronizing while connected via the Internet. I recommend that you then add these macros to the Taskbar for this UserID. Test out both of the buttons to make sure that they work properly, and that should be it.

I hope that your Remote-side has synchronized successfully to your Server-side, however, even though GoldMine is one of the best products for synchronizing, the synchronization can be bad or fail for many reasons. David Lee is allowing me to reprint his whitepaper on this subject, which I originally published in The GoldMine Advisor newsletter.

Whitepaper

Note

David Lee
Vertical Marketing, Inc.
10021 Balls Ford Road
2nd Floor
Manassas, VA 20109
USA

(703)367-9571 2222
DLee@VerMar.com
http://www.VerMar.com

This whitepaper was written by David Lee, and with his permission is duplicated here for your edification unaltered..

GoldSync Problems

How does GoldMine sync work compared to other systems?

The major CRM systems have each found its own way to synchronize data. None of them are perfect, and all of them will lose data in some situations. Despite this, the value of being able to synchronize remote systems, and work offline is so great that virtually every major CRM system provides this option (including web based ASP systems). You would like to find a system that loses data rarely, that allows you to find out whether or not you have lost data, and that gives you a good way to recover the data.

Some systems, such as SalesLogix, dynamically build transfer sets as you enter data. This is an effective approach when you have a very small number of syncers. However, it is very slow if you have large numbers or if there are problems which require you to rebuild your transfer sets.

Other systems, such as Microsoft CRM, track changes and force an update when you dock and undock from the system. This gives you little or no ability to correct sync errors or reset your sync. In Microsoft's case, it also lacks sophisticated conflict resolution rules. What do you do if two people updated the same field, and then you Sync the systems?

GoldMine builds the transfer sets when you ask for them to or it pre-builds them on a schedule. The building process is resource-intensive, but it can support a very large number of users without degrading system performance. GoldMine also has a sophisticated conflict resolution capability (the most recent change wins with proper adjustments made for users in different time zones). In addition, GoldMine creates transaction logs (Tlogs) to track who made changes, when they were made, what was changed, and when the change was synchronized. This ability to create transfer sets at any time gives GoldMine great flexibility, and scalability compared to other systems.

It also creates many vulnerabilities as discussed in this white paper.

Why is it better? Why is it worse?

- **Advantages of GoldSync**
  - Scalable to large numbers of users
  - Supports many topologies, and methodologies
  - Very flexible in determining what will, and will not sync
  - 1-button sync feature reduces users' ability to introduce errors
  - Easy (compared to other systems) to run a "catch up" sync when there are problems
  - Good conflict resolution algorithm
  - Supports territory realignment
  - Allows single record synchronization (for example, you can synch a single record to quickly transfer an account to another person)
  - Allows lateral synchronization (synchronization between users as well as synchronization with a master database). This adds great flexibility, and power.
- **Disadvantages of GoldSync**
  - Difficult to set up
  - Users can introduce errors
  - Difficult to diagnose errors (this is common to most synchronization algorithms, not just GoldMine)
  - Transaction logs add significant complexity to resolving certain classes of problems

- Conflict resolution is not perfect. There is no ability to configure a rule, for example, that a manager's changes will override a sales rep's changes.
- Does not have the concept of "inherited rights," in which a manager can elect to receive any data that his subordinate can receive during sync.
- Allows lateral synchronization (synchronization between users as well as synchronization with a master database). This provides additional opportunities for user errors.

How do I detect sync problems?

One of the problems with synchronization is that it is often difficult to detect problems. How do you know that you did not receive a particular transaction or update? You may want to run a monthly report (using a reporting tool, a SQL query, or other method) to compare summary data from all GoldMine systems. If the totals do not balance, this might indicate sync problems.

■ Use the GoldMine Statistical Analysis to compare summary data between different GoldMine databases.

■ SQL queries, sync logs (if activated), and e-mail alerting tools can help determine if there are potential problems. For example, you can develop a series of SQL queries to let you compare summary data between GoldMine databases.

■ Also, Sync Spy (Tools|Sync Spy) can tell you when, and by whom records were last modified right down to the FIELD level.

■ Observation: Users report missing or inconsistent data. In practice, this is by far the most common way that sync problems are discovered.

How do sync problems happen?

■ **Errors in sync filters**

- The sync filter may have been set up incorrectly. Some filters are quite complex, and can have faulty logic.
- The logic of the filter may correctly reflect the original specification, but the specification may not give the intended result (for example, I want to sync only records "owned" by a sales rep, but I forgot to sync records common to the whole team. Therefore, the rep feels that he is missing records. He can sync up a team record that he creates, but never gets updates from the sync server. He views this as a sync error.).
- User-introduced sync filter errors. Sometimes users click into the sync area just to see what is going on. They may change a setting or actually load a different filter accidentally.

■ **Errors in sync settings**

- In addition to filters, there are many other settings governing what data is sent, and what data is accepted during a sync.
- Some settings may have been made in error (for example, settings that prevent syncing all calendar records when the actual intention was to prevent syncing calendar records that do not belong to the user but to allow syncing records that do belong to the user).
- Some settings may have been changed by users who are "Browsing" through the sync settings.
- It may be difficult to identify these changes, especially if they have been corrected by the time the GoldSync engineer starts to investigate the cause of sync problems.

■ **Inappropriate Record Settings**

- You can mark individual documents, and records to NOT sync. This is very valuable when the documents are personal, or when very large documents would require excessive transmission times. However, users may forget that they flagged the document or transaction not to sync, and then view the failure to sync as an error.
- GoldMine has a calendar "rollover" feature to automatically forward old, uncompleted activities to the current day. The rollover could involve large numbers of transactions, and there is an option to flag them as a group to not sync. Again, a user could forget that he made this setting, and view the failure to sync as an error.

■ **Misunderstanding how deletions sync (delete filters vs. deletion logs).**

- When you delete a record, you create a transaction log entry. This entry synchronizes, and causes deletion of the record in the synchronizing computers. The record can never be restored as long as the log exists.

• When you create a deletion filter or a territory realignment, there is a different type of deletion record. It does not remove data from computers other than the target computer, and it does allow records to be restored.

• If you move a record from to another file (such as an archival file), it creates the second type of record. This will not cause the record to be deleted from other systems during sync. This may be incorrectly viewed as an “error” by some users.

• If you want the record deleted from a target computer but not from other computers (such as in a territory realignment), you must create something called a “deletion filter”.

• A user may reassign a record to another user by changing the record owner field or by updating a sales rep field. If his own sync filter does not allow him to see records with the new setting, he will no longer get updates for those records. Moreover, if nobody creates a deletion filter for him, the record will not be removed from his computer. Finally if he gets frustrated, and deletes his record, he might delete the record everywhere in the system. All of these could be viewed as sync errors, but, in fact, all can be addressed by proper sync design, and sync settings (such as not accepting deletions from the main sync server).

■ **Using the wrong default sync settings (not restoring the default)**

• GoldMine allows many synchronization options. For example, it is the only major CRM system that allows two users to sync to one another, and at the same time sync to a primary sync server. Each sync configuration that a user defines can be saved as a “sync profile”, and a user can have as many such profiles as he desires.

• The default profile is always the last one that was used. If a user selects or defines a new profile for a 1-time use or just because he is browsing around the system, that will become the default.

• If the user does not remember to restore the original default, all future syncs (until he realizes the problem, and restores the correct profile) will produce an unplanned result. For example, he may build a new sync file (called a Transfer Set) over, and over without actually completing a synchronization. If he realizes his mistake, and restores the original settings, it will be difficult for the GoldSync engineer to identify the cause of the problem.

■ **Database / index corruption**

• Many laptop systems (and some network systems) use the dBase database structure rather than MS SQL. DBase uses indexes to associate the GoldMine files, and these indexes are subject to corruption. If the corruption is not corrected (using the reindex or rebuild tools), the system could start losing track of the relationship of the files that build the record. This could not only cause synchronization problems, but the corruption could even be passed to the server, and then from the server to other laptops.

• There could be other database corruption problems such as BLOB (Binary Large Object, or Notes fields) errors. Very large note fields are especially vulnerable to this type of corruption. The best preventative measure is to keep many small notes (such as separate notes for each call or meeting) rather than one very large notes field for several interactions.

■ **Duplicate CONTACT2 records**

• This is a specific kind of data corruption that is especially insidious. Each Contact1 record expects to have a single Contact2 associated record. For a number of reasons, including index corruption or someone restoring a system by overlaying a file, GoldMine may not realize that it has a Contact2 record, and may build another one. This creates duplicate Contact2 records.

• When you look at a record or sync a record, you will randomly select one of the Contact2 records to associate with the Contact1 record. Depending on which one you see, some fields will have missing or incorrect data. Next time, if you happen to see the other Contact2 record, you will see different values in the fields.

• Since this problem will synchronize, the resulting issues may be viewed as sync problems.

■ **Deleting transaction logs**

• If possible NEVER purge the Tlogs, although this is highly unlikely as they will bloat in size, and cause the system to slow down. If you do purge the logs, make sure that you first create a transfer-set of the GoldMine customizations. This is the only way you can send customization changes to remote sites. Also, save a few months worth of TLOGS rather than purging ALL Tlogs

- From time to time, it may become necessary to delete your transaction logs or individual log entries. One reason might be that someone has wrongfully deleted a record, and you must delete the offending deletion log entries in order to restore the record.
- The problem is that this is a very difficult process, and people tend to do it incorrectly. Often they delete entire files when they should be deleting individual entries.
- When the file is deleted, GoldMine “forgets” information critical to synchronization, and “errors” could result.
- There are transaction logs in the main sync server, and in all synchronizing computers so there are many potential points where errors could occur. This makes it difficult to identify the problem.
- The protocols necessary to ensure that you do not lose data when you delete the transaction logs are difficult to follow, and few people do it correctly. The protocols require that every synchronizing computer in the entire environment be involved, and usually all in the same day.

■ **Syncing with a filter active**

- A user may activate a group or filter for any number of very valid reasons. However, it he accidentally leaves it active when he syncs, then only the records defined by the filter or group will sync.
- Note that if a user has active tagged records or has activated an org chart section, it has the same effect as an active filter.
- Next time he syncs, the sync will be from the date, and time of the last sync, so the records that were missed will not be synced.

■ **Syncing while logged in as someone else**

- Sometimes users log in as someone else for a number of reasons (for example, they may log in as “Master” in order to perform a function that requires master rights).
- If they forget who they are, and sync, they will be using a wrong sync filter. In the case of Master, it is probably no filter at all which will mean that they are syncing all of their records. In addition, Master will be syncing as of the last time he synced (which was probably “never” in the case of a laptop) so it could sync years of data, and cause havoc in the main system. Again, proper setup could prevent this or at least make it extremely difficult for the user to make the error.

■ **Incompatibility between the GoldSync Server, and the Client**

- The server, and the remote sites should always use the same encryption level.
- The server, and the remote sites should always be on the same version and build.
- Make sure that the remotes Regional Settings are appropriately selected for where that individual may be based (GoldSync automatically compensates for time differences in different time Zones).

■ **Locking an active record**

- If you have a record open in edit mode (including a history or calendar record), it is locked, and will not sync. Over time, this could cause an accumulation of unsynced data.

■ **Locking a document**

- If you have a document open, then it is locked, and will not sync.

■ **Syncing with the wrong database open**

- You may work on multiple databases (such as a primary, and an archival database). If you are in the archive, and forget where you are, you might launch a synchronization from that file. This will cause the wrong data to synchronize. In some countries it is legal for the sync administrator to shoot you if you do this.

■ **Duplicate Record Merging**

- When you merge duplicates you might merge two records representing the same person, but which were each “owned” by different reps. After the merge, the record may “disappear” from one rep, who will view it as a sync error.

■ **Overlaying files**

- In an effort to restore data or to correct other problems, a GoldMine Administrator may copy a file such as the Contact1 file or the Calendar file for dBase systems. This can be

a very efficient shortcut to correct problems. However, if it is not done exactly correctly it can cause extensive data corruption, and/or sync corruption.

- DO NOT "selectively" restore Tlogs from a backup. Doing so may cause recently entered records to be tracked incorrectly!

■ **Database structure changes**

- You may have changed the field structure in GoldMine. Assuming that your sync settings allow, these changes will go to your remote sites through sync.
- If the same sync set that changes the field structure also contains data for these new fields, that data may not sync because at the time of the sync the fields do not exist.

■ **Transmission and application errors**

- The sync files may be corrupted during transmission. If they are not received at all, then GoldSync will re-send them in the next sync session so there will be no long term harm. However, if they are received, and the sync process starts, and if it is aborted because of an incomplete file or other transmission problem, GoldSync will think that it got the file, and will not realize that data is missing.
- If GoldMine or Windows freezes while the sync set is being applied, or if there are other application errors, the data that is not applied will be lost. GoldSync will not realize that not all data has been applied.

■ **Out of space**

- If you run out of disk space during a sync, that sync will abort.
- If you are not paying attention to the progress monitor, you may not realize that you had a problem.
- In most circumstances, GoldMine will realize that the sync did not complete properly, and will "catch up" the next time that it is able to run a complete sync.

■ **No Transfer Set to Pick Up**

- In one of the sync options called queued processing, transfer sets are created on a schedule, and are then sent when the user syncs. This saves him the time of waiting for the transfer set to be created.
- In the event that there is no transfer set available, he will get a message in his progress monitor saying that there is no transfer set to pick up. If he does not read the message, he may not realize that he has not gotten any data, and he may view the lack of new data as a sync error.
- In most circumstances, GoldMine will realize that the sync did not complete properly, and will "catch up" the next time that it is able to run a complete sync.

■ **Restoring a database from a backup**

- Sometimes a GoldMine database must be restored from backup files. This may be the primary database or it may be a laptop database.
- Since the database, including all sync logs, has now been restored to an earlier version, some data could fail to sync.
- Even worse, the database administrator may only restore selected files. Depending on the sync rules in force, and the files that were restored, this could cause any number of problems with synchronization. If this happens to you, try your best to find out when the files were restored, what files were restored, and the date(s) of the restored files. Give this information to the person who is helping you to recover.
- This can create some very complex problems, especially if there have been several synchronizations after the backup restore, and before you realize that there is a problem.

■ **Main contact record Notes tab**

- When you enter data into the Notes tab in the main GoldMine record (not the Notes of a calendar or history or Detail tab transaction), that note is automatically date/time stamped.
- GoldMine will synchronize the new note on the next sync, however, if it sees the same date/time stamp in the receiving system, then it will assume that it has already been synced, and will not apply the new note.
- This means that if you edit such a note after you sync, the edit will never sync (even if you set back the sync clock). This is not a bug: It is the way the system is designed. Nevertheless, many users will view it as a sync problem. The solution is to add new notes when you need to change what you said, rather than editing old notes.

■ **Program bugs**

- From time to time we find actual program bugs that could affect sync. These could be bugs that have always existed, but occurred so rarely that nobody ever noticed, or new bugs introduced when some part of GoldMine was updated.
- FrontRange will correct these bugs if they are reported, and if they can replicate the problem.

How do I correct sync problems?

■ **Attempt to diagnose the problem**

- Bearing in mind that the problem could have many causes as noted above, review the sync logs, inspect the files, and otherwise do what detective work you can to determine the cause of the problem. This is not an easy task.
- Use diagnostic tools to help identify the problem. A tool called “Sync Spy” is built into GoldMine. It will show you the “Sync Stamp” of various Contact1, and Contact2 fields.
- If you are unable to diagnose the problem, you can often cure the sync problems without even knowing what caused them using the following techniques

■ **Set back the sync date**

- Manually set the From date, and time to a time earlier than the last date when you are confident that all data was properly synced.
- Ensure that you have not locked any records or documents, that you have the proper sync settings, etc.
- Run synchronization to “pick up” individual updates that were missed in prior synchronizations.
- Be aware that, depending on where the data is missing, you may have to set back the sync dates on the sync server, on the remote system(s), or both. In most cases it is safest to set back the date on all systems.

■ **Correct data problems**

- Pack/rebuild your files.
- Use dSalvage or other utility to find and fix BLOB errors (as mentioned earlier, BLOBs are Binary Large OBjects). In GoldMine, BLOBS are the Notes fields. If a process locks with a BLOB error on a particular transaction, then that transaction (or the one following it) is probably the offending record. An easy cure is to:

  Open GoldMine
  Go to that record
  Edit the record and copy the notes onto your Windows NotePad
  Delete the note from the transaction
  Save the change
  Open the record in edit mode
  Paste the notes back in from your Windows NotePad
  At this point, the BLOB error should be cured

- Take other steps as necessary depending on the nature of the corruption: Any database can become corrupted in dozens of ways. The corruption will cause sync problems. It is beyond the scope of this white paper to discuss all possible forms of database corruption.

■ **Delete transaction log entries**

- Identify the exact RecID of the wrongfully deleted record(s).
- Find, and delete the corresponding deletion transactions in the transaction log of every computer that was affected.
- WARNING: If you miss a single computer, and if that computer later syncs, you may need to repeat the entire process. To prevent this problem, you should “turn off” sync in the GoldSync Administrator for all remote databases, and “turn on” each only after you have confirmed that it is “clean.”.
- This is difficult, but far less difficult than the protocol for deleting the entire transaction log.

■ **Delete entire transaction log**

- All remote computers, and the sync server must be backed up.

Note

This Whitepaper was written when GoldMine still had a Standard Edition brother that ran on dBase, hence, Davids recommendation of dSalvage. In GoldMine Premium you will not see BLOB errors or be able to utilize dSalvage to fix anything.

- All remote computers must sync all of their data to the sync server. There should be no transmission sync filters.
- There must be no further entries after this sync.
- Transaction logs must be deleted from all remote computers, and from the sync server.
- The sync server sends deletion filters as needed.
- The sync server sends a transfer set dated as far back in time as necessary (depending on your analysis of the data problems).
- This is a very difficult protocol in practice. If even one remote computer does not follow the protocol then the problems could be reintroduced.

■ **Full sync restore**

- All remote computers, and the sync server must be backed up.
- All remote computers must sync all of their data to the sync server. There should be no transmission sync filters.
- Delete all data, and all transaction logs from all of the laptops.
- Set back the sync date of the sync server to day zero, and build transfer sets for each remote computer.
- Completely rebuild every remote computer using the new transfer sets.

15

Leads Management Center

In This Chapter

The GoldMine Leads Management Center helps you manage your leads at different stages throughout the sales cycle. The Leads Management Center provides a set of tools that assist you in defining ownership and security rules, analyzing your leads, and more. This statement, taken from the header of GoldMines' **Leads Management Center**, pretty much says it all. This Center is basically a grouping of all of the GoldMine Tools, which can also be accessed elsewhere, that will assist you with your corporate Leads Management.

From the GoldMine menu select:

Go To
Lead Management

This will access the GoldMine's Leads Management Center, the resulting dialog form can be seen on the next page in full, Figure 15-2. Basically, this center is a combination of tools within a single GUI. Some of these access points are to previously existing tools within GoldMine, while others access totally new tools that had been requested by users for a long time. FrontRange has listened, and delivered on your requests.

I will get to each of these in turn, however, I would like to mention that some, like **Assign a Merge Code**, have been sought after for years by end users of GoldMine. I wrote an application to do just this for a client about eight years ago. It is great to see that these tools have now been incorporated into the GoldMine application.

I begin by asking you to look at Figure 15-2, next page, to see the GoldMine **Leads Management Center** which, in GoldMine Premium, has been demoted to just the **Leads** dialog form.

Import Leads into GoldMine

The first tool, **Import leads into GoldMine**, is another way to access the GoldMine Import tool which has existed, in various versions, for years. The only thing new for GoldMine Premium, when you click the hotlink Import leads, is that you get to predetermine the type of import that you will be using as shown here, Figure 15-1, in the **Select Import Type** dialog form.

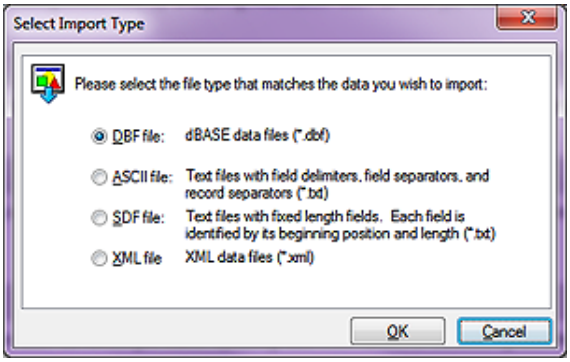


Figure 15-1

Selecting any of the first three import options will bring up the standard Import dialog form as shown ahead in Figure 15-3. Selecting the last entry, ○ **XML File XML data files (*.xml)**, will bring up the XML import dialog form, which is different. As I will not actually be discussing the actual imports themselves, I will not be showing you any more dialog forms driven from this utility.

All that the user is required to do, is to select the data type that is to be imported, and then to click on the **OK** button. From then on, you, the user, is presented with the normal standard GoldMine Import utilities with the data type preselected.

Figure 15-2

Figure 15-3

Assign a Source Code

WARNING

This is a destructive utility. Before attempting to use this utility, one should always make a backup of their GoldMine. This is always the case when performing any destructive actions within your GoldMine.

"Not a great tool.", you say. Well, yes, and no. I like the fact that GoldMine consolidates these tools into an easily accessible area. Having said that, let's continue on.

Assign a Source Code is a slight misnomer. The user should consider this tool as a mini **Global Replace...** Wizard, and, as with the regular **Global Replace...**, the user should always backup prior to employing this utility.

The user can easily see, Figure 15-4 on the next page, that they can employ any of the available GoldMine **Filters/Groups**. I have employed my filter called the GoldMine Premium Filter, as I, myself, may end up using the **Campaigns** tool within

Note

*There are the 10 user defined fields that are not true user defined fields, **Contact2.UserDef01** thru **Contact2.UserDef10**. You will have to examine the list carefully. If a user defined field has been given a **Global Label**, it may appear in the list with the Global Label as well as the user defined name. On the other hand, it could simply appear in the list in alphabetical order based on the user defined field name.*

Note

*Did you notice that they said ☐ **Update linked fields (based on Lookup.ini)**. Fields is plural, whereas, you can only select a single field using this utility. This is a carry over (code reuse) from the **Global Replace** utility.*

GoldMine. I will definitely want to send all of my current readers notification of the release of this new book.

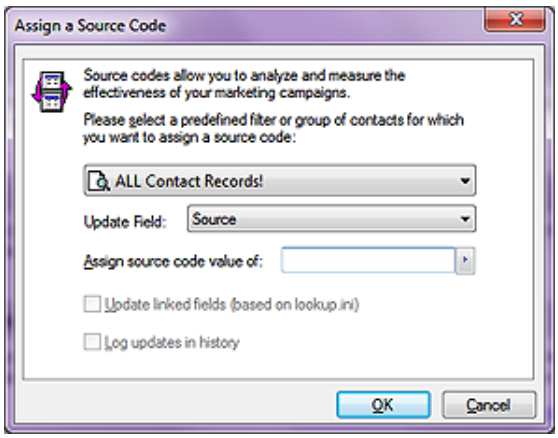


Figure 15-4

Next is the **Update Field:** drop list box (have you noticed that there is no hot-key for this field?). Even though this utility is entitled **Assign a Source Code**, you may select any **Contact1**/**Contact2** field to replace, hence the previously mention misnomer. The drop list will display all of the fields in alpha order using the local labels except for any real user defined fields, which are listed by their user defined field name. The user simply has to select the field for which they wish to replace any of the existing data with the new data. Unlike the real Global Replace utility tool, you may not use expressions here. You must select a field from the provided list, and you may only do a single field per run, whereas, on a true Global Replace, you could replace multiple fields on the same run.

Assign source code value of: is the replacement information. Should the previously selected field have a **F2 Lookup List** then that list will be available for entry selection. Usually, as this is **Leads Management**, the entry will not pre-exist on the **F2 Lookup List**, and you will need to enter it by hand.

The next two items are grayed out by default in most cases, however, should you select to replace a field that is contained as a watch field (left hand side of the equal sign) within your **Lookup.ini**, or that has been registered to create a **History Activity**, then the appropriate option will be enabled, and available for selection. ☐ **Update linked fields (based on lookup.ini)**, and ☐ **Log updates in history** are these two options of course. Even though the field may be marked to create a History Activity, the default selection here will be to not create a History Activity, and rightly so as this is, in effect, a Global Replace.

If you have already made your backup of your GoldMine, from here it is a simple matter of clicking on the **OK** button, and the processing will proceed. I really mean that as GoldMine provides no intermediate dialog "Are you sure" question. Once you click on that **OK** button, you are committed. Shoot, it is so simple, that I think that even I would be successful in doing this easy task.

This is an important utility when used in conjunction with your Leads Management. It is very important to be able to group your leads into a manageable segment of your database. Updating a Source field is important in this process, however, have you noticed, you must be able to group the leads already in order to apply this process? This is kind of a Catch 22, however, I'm sure that you'll be able to figure other uses for this utility as well.

Analyze Leads

To **Analyze Leads** is an old utility presented to you through the **Leads Management** interface. Look here at Figure 15-5, and then Figure 15-6 on the next page. You may recognize these GUIs. Figure 15-5 is used to create the file into which the result analysis is placed, while Figure 15-6 is actually used to populate that file.

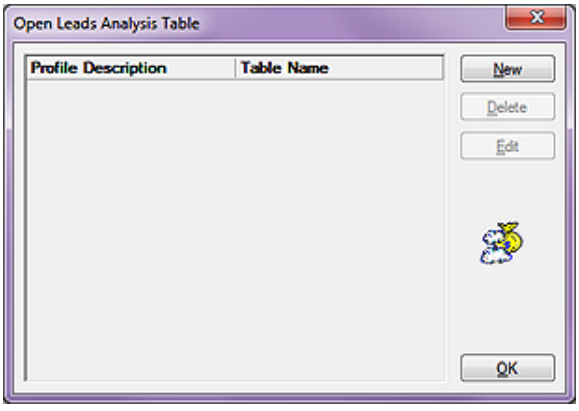


Figure 15-5

You should notice that this utility, and the last utility, go hand in hand. I can run a **Leads Analysis** against any field, while the **Assign a Source Code** lets you assign a **Source Code** to any field. It is very nice of FrontRange to group these sequential utilities into an easily accessed location. I won't actually discuss the **Leads Analysis** dialog form, as it has been around for a long time now. When you click on the **Analyze** button as shown in Figure 15-6 on the next page, you will be presented with a dialog form from which you may choose your field to analyze against, as well as a date range to analyze within.

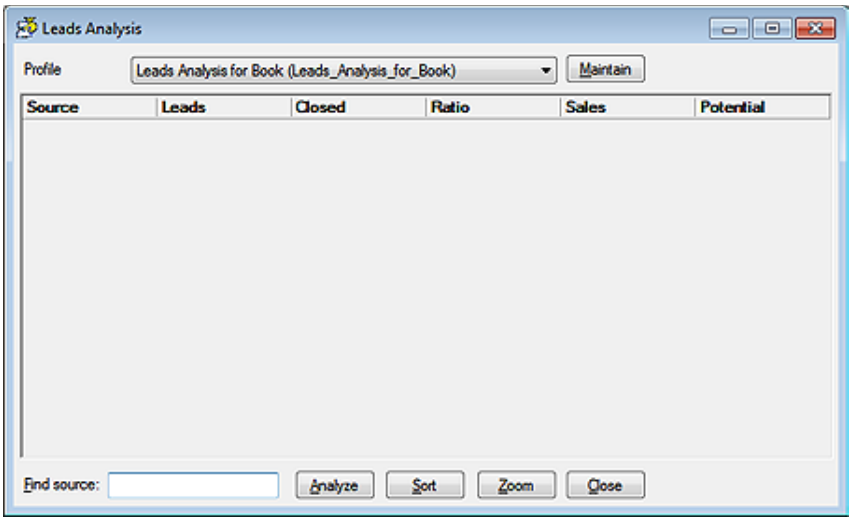


Figure 15-6

Assign a MergeCode

WARNING

*As of this writing, this utility is a **replacement** (overwrite) utility. Using this utility will replace everything in the **Contact1.MergeCodes**/**ContSupp.MergeCodes** field with whatever value you enter. I have requested, from FrontRange, that they change this to an append utility, or allow the use of dBase functions.*

Where this is a destructive procedure, it would benefit you to have a SQL Backup of your GoldMine Premium Database before your begin this procedure.

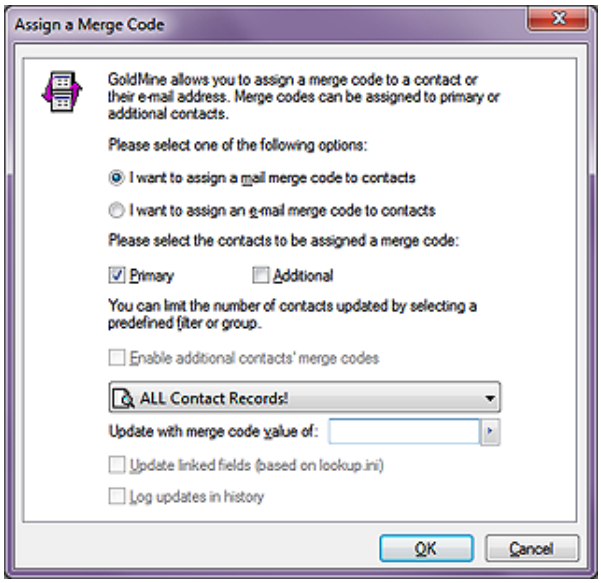


Figure 15-7

Note

*I would like to reiterate that a mail merge code pertains solely to **Word Document Merges**, while an e-mail merge code pertains solely to Internet **E-mail Merges**.*

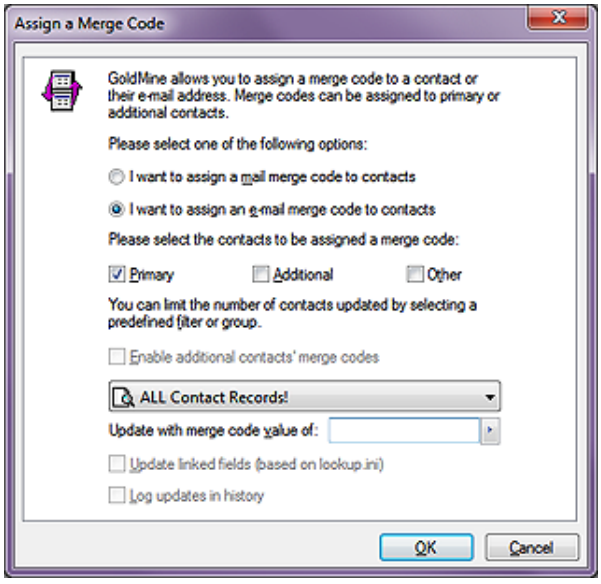


Figure 15-8

Well here is a long awaited for, and often requested utility as GoldMine states it: "*Merge code values allow you to better coordinate your mail and e-mail merge efforts. To assign a merge code value to your leads, click Assign a merge code.*", and that's a fact Jack. Individuals have a hard time comprehending that there are two separate fields for **MergeCodes**, and that each is employed independently of the other. Everyone sees the **Merge:** field on the primary GoldMine screen, and usually enters a code there while not realizing that this merge code field is for document merges for the primary contact only at least within the default GoldMine practices. It has absolutely nothing at all to do with **E-mail MergeCodes**. Unless they access the **ContSupp** table (display the E-mail Addresses by clicking on the E-mail Address from the primary GoldMine screen), they never see that there is a separate, and distinct, **Merge codes:** field for the e-mail addresses (each E-mail Address listed has its own **MergeCodes** field). Additionally, each secondary contact, under the **Additional Contacts** tab, has its own document **Contact1.MergeCodes**, and E-mail Address **ContSupp.MergeCodes** field. Wow! Now I know why individuals can easily become confused. I think I just confused myself.

In simple terms, there is a **MergeCodes** field for documents for each Primary, and Secondary Contact in GoldMine. There is a **MergeCodes** field for each individual E-mail Address within GoldMine. This utility allows you to overwrite the **MergeCodes** fields any where they exist. The key words in that statement being "overwrite the **MergeCodes** fields".

You have two option selections on this dialog form that determines how the dialog forms look, and feel. The default option is ⦿ **I want to assign a mail merge code to contacts**, and Figure 15-7 on the previous page depicts this dialog form. If, however, the user were to select ○ **I want to assign an e-mail merge code to contacts**, then Figure 15-8 on the previous page depicts the respective resulting dialog form. One can easily see that the only difference between the two dialog forms is an additional option, ☐ **Other** for when one selects the E-mail Address selection option. This is because the primary Contacts may have additional E-mail Addresses. They are not limited to one Primary E-mail Address, and you may wish to assign a merge code to each of the E-mail Addresses. On the other hand, the secondary Contacts may only have a single E-mail Address each.

I will discuss the non default dialog form selection, as shown in Figure 15-8, as this is the feature that was lacking before the introduction of GoldMine Premium. Next you must choose among the options presented to select the contacts to be assigned a merge code. By default the ☑ **Primary** is selected, and you, the user, may, in addition, select ☐ **Additional** and/or ☐ **Other** (remember the ☐ **Other** option will not be available for the document (mail) merge code).

Next *"You can limit the number of contacts updated by selecting a predefined filter or group"*. There is a drop list available from which one may select the filter/group that they wish to apply this replacement against. This is good if you have a Filter/Group created for this lead group. You may have noticed a grayed out option ☐ **Enable additional contacts' merge code**. Additional or Secondary Contacts have a flag preceding the **Merge:** field which enables or disables the **ContSupp.MergeCodes** field for this particular secondary contact. If you have selected ☐ **Additional** on Figure 15-7, this option will be enabled, and you should then select the option. In fact, I am not sure why this is not selected when enabled as a default. I should mention that the Filter/Group drop list acts as all similar list throughout GoldMine. If you have a Filter/Group Active, then the list will default to **Active Contact Filter**, otherwise, it will default to **All Contact Records!**. You may also select any Filter/Group available to you from the drop list.

Now we come to **Update with merge code value of:** and a click on the F2 Lookup List button will bring up the F2 Lookup List for the selected merge code field. Alternatively, you can just type in the replacement merge code value, remembering that this will replace any existing merge code information for the selected records. This is not, as it implies, an update for the field. It is, in fact, an overwrite procedure.

The next two items are grayed out by default, however, should you select to replace a field that is contained in your watched **Lookup.ini** fields, or that has been registered to create a **History Activity**, then the appropriate option will be enabled, and available for your selection. ☐ **Update linked fields (based on lookup.ini)**, and ☐ **Log updates in history** are the two options. Even though the field may be marked to create a History Activity, the default selection here is to not create a History Activity.

I'm pretty sure that you know to click the **OK** button next.

Assign an Owner/Manager

Most organizations will **Assign an Owner/Manager** to take control of a new lead or group of leads to assure that none falls through the cracks, and all process appropriately for their organization. As FrontRange states it: *"Record ownership helps you take control of your leads by defining who can update their information, and how much of their information others can see."*

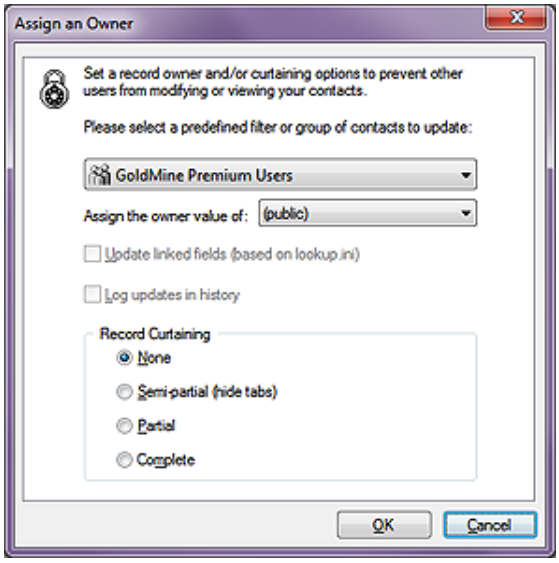


Figure 15-9

This process itself is quite simple, and again, you would like to activate a Filter/Group to which you may **Assign an Owner/Manager**. In the first section of Figure 15-9, I used my sample filter: **GoldMine Premium Users**. Even though these are not new leads, when next I look at the GoldMine **Campaign Management Center**, I will want to try an up sell this group to **GoldMine Premium - The Definitive Guide**. Who knows, you may be in this category, and have already become part of this marketing example.

Next, it is assumed that you have already defined your **User Groups**, and are capable of selecting the appropriate **Group/UserID** from the list of those available under the **Assign the owner value of:**.

Again, this is a drop list, and not a combo list. You may only select from, and you may not type into it (if you type, the list just advances to a matching item if one is available).

And next we see our two friends again. These two items are grayed out by default, however, should you select to replace a field that is contained in your watched Lookup.ini fields, or that has been registered to create a History Activity, then the appropriate option will be enabled and available for selection. ☐ **Update linked fields (based on lookup.ini)**, and ☐ **Log updates in history** are the two options. Even though the field may be marked to create a History Activity, the default selection here is to not create a History Activity. Boy, where have we heard that one before?

And, lastly, there is the **Record Curtaining** frame. These are, as always, radio button selections (only one can be selected). Your options are:

- ◉ **None**
- ○ **Semi-partial (hide tabs)**
- ○ **Partial**
- ○ **Complete**

For those of you who don't remember from Chapter 9 - The Tables, the values represented are:

0 = None (legacy)
1 = Partial (legacy)
2 = Full (legacy)
3 = Semi-Partial (recent GoldMine versions)

Those values are stored in the 2nd bit of the **Contact1.Status** field while the Owner/Manager Owner Group/UserID are stored in the **Contact1.Owner** field.

And that pretty much finishes it for this dialog form. Simply click on the **OK** button to proceed with the replacement.

Assign an Automated Process

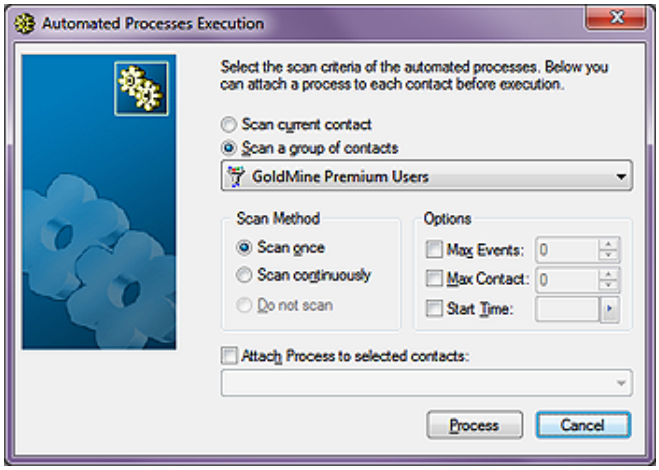


Figure 15-10

Again, we have an old tool presented to you via a different GUI. There is very little to discuss here, however, for those that have never done this, I would like to say that you must first have your **Automated Process** defined. Further, as you can see, I will be applying it against my default filter, **GoldMine Premium Users** in keeping with the spirit of the campaign. What you don't see here, in Figure 15-10, is that I have yet to check the option ☐ **Attach Track to selected contacts:**, and this must be done before the **Process** button is clicked upon. Once checked, all available Automated Processes will be available in the drop list from which you may choose the appropriate new leads process to be assigned to the campaign group.

Schedule an Activity

Again, we have an old tool presented to us via a different GUI, and as before this interface should be no challenge to the long time GoldMine user. You may have noticed, Figure 15-2, that you could select the **Activity Type** directly from the **Lead Management Center** dialog form. For Instance: Figure 15-11 on the next page shows that I had selected to **Schedule a(n) Next Action**, as the **Schedule a Next Action** dialog form was driven from the Leads Management Center dialog form.

You would complete the **Detail** tab dialog form, and then click on the **Search** button to the right of the ☑ **Link to selected Contact:** field and choose **Select Filter or Group...** to continue the process of scheduling for your specific lead campaign group.

Organize Filters or Groups

This is another GoldMine faux pas, as far as I am concerned. This is a good tool to have in this the **Leads Management Center**, but wouldn't you have thought, since we have used **Filters/Groups** in every tool so far, that this would have been the first item in the **Leads Management Center**? Especially where the FrontRange developers didn't even follow an alphabetized presentation of these tools for the user.

Oh well, everyone has their own opinions as to User Friendliness when it comes to application design.

As FrontRange states: *"Filters and groups help you quickly access your leads and segment your database on your own criteria. To organize and work with your filters or groups, click Organize filters or Organize groups."*, and I recommend that you try to establish a filter or a group as your first step in this whole Leads Management process. Much as I had done by pre creating the filter: **GoldMine Premium Users**.

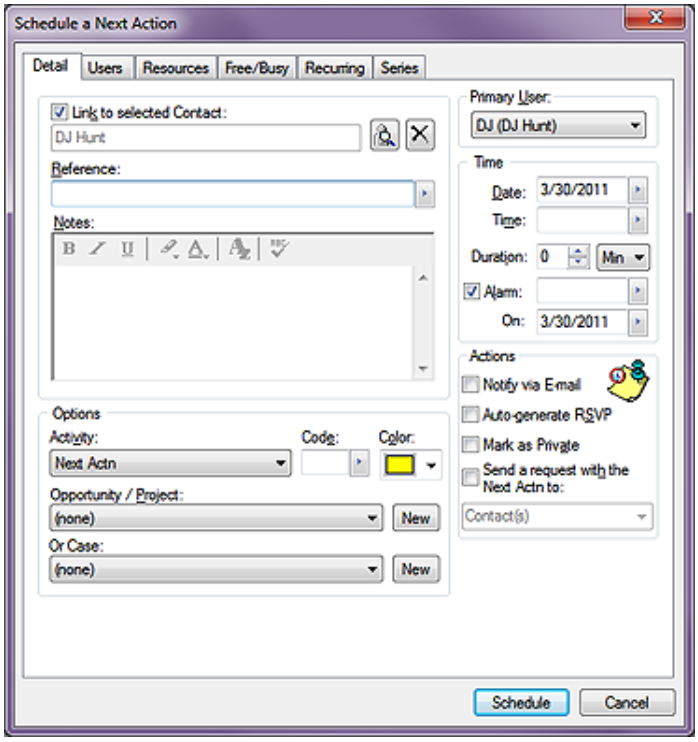


Figure 15-11

As this selection only brings you to the **Filters and Groups** dialog form, and having already discussed these in detail in Chapter 7 - Gathering the Data, I do not feel a need to discuss these any further in this chapter.

This, then, concludes this chapter on the **Leads Management Center**.

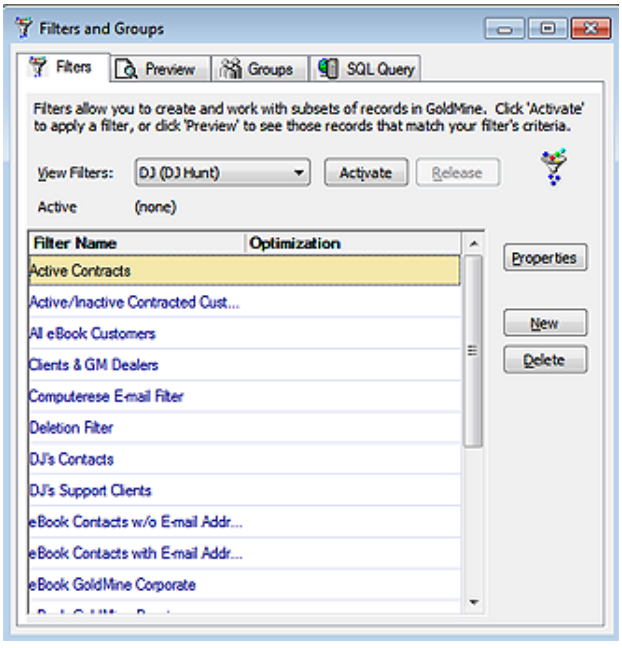


Figure 15-12

16

In This Chapter

The Opportunity Manager

The GoldMine **Opportunity Manager** is a monster unto itself. Until the last Hacker's Guide I had refrained from presenting the Opportunity Manager in this series of books. The Opportunity Manager consists of, and displays information from a number of tables in a consolidated GUI. The problem being that there are few really good reports created for the Opportunity Manager within GoldMine and even fewer Dashboards, one exactly. It is darn near impossible for an end user to create a report from the eight tables utilizing the GoldMine reporting tool, and only a slightly easier capability, because of the pre created categories, to build Dashboards. Therefore, those that need a hard copy from the information contained within the Opportunity Manager are pretty much up a creek without a paddle, so to speak, or at least on their own.

If, however, the information displayed in the GUI is sufficient for your needs, then the Opportunity Manager is a great tool for consolidating your opportunities under a single GUI, although that single Dashboard does a great job of emulating this display. Though I won't be discussing it in to much detail in this chapter, the above statements also apply to the **Project Manager** as well. You see, the Project Manager uses the exact same tables as the Opportunity Manager, and most often an opportunity is simply rolled over into a project as the opportunity closes.

I begin at the beginning with the Opportunity Manager GUI as depicted on the next page in Figure 16-1. Access to this GUI is achieved by selecting **F9** from your keyboard or...

GoTo
Opportunities

...from the GoldMine Premium menu. This GUI displays required information concerning your opportunities. You may view **Influencers** which are related to and hyperlinked to **Contact1** records within your GoldMine database. Then you have **Prods/Svcs** (Products/Services) which are nothing more than a **Forecast Sale** from the GoldMine **Cal** table. The **Prods/Svcs** information is also repeated under the **Pending** tab in this GUI. Next you have the **Tasks** tab where one can create a time line for the opportunity assigning tasks to various **UserID**s. This data will be stored in the **OpMgr** table using a **RecType** of **OK**. The graphical display of the time line is rather neat, and may be color coded as, say for, a Critical Priority that you may want graphically displayed in **Red** as opposed to the default **Blue**. You may then select your **Team** members, and they may be either a User, a Contact or both from within your GoldMine. These bits of information will be stored in the **OpMgr** table with a **RecType** of **OT**, and will contain an **AccountNo** for a linked **Contact1** record, or be blank if the member is that of a GoldMine **UserID**. One can also have **Issues** which may effect the opportunity, and these will again be stored in the **OpMgr** table with a **RecType** of **OI** this time. This **OpMgr** table is starting to look suspiciously like the GoldMine Premium **ContSupp** table wouldn't you say? One table being used for many records differentiated only by record type. Now where have we heard that one before?

Let's finish our quick overview of the various tabs. I will discuss these in more detail later in this chapter. The next tab is the **Notes** tab, and, like all notes, is limited, however, in GoldMine Premium you should never realize these limitations. These notes are stored in the primary opportunity record in the **Notes** field (go figure). Next you could add those horrible **Competitors** that you wish to keep abreast of. These can be hand typed, or selected from the **F2 Lookup List** if you have many repetitive competitors. Additionally, they could be contacts in your **Contact1** table. Again, GoldMine employs the **OpMgr** table for this task with a **RecType** of **OP**. The **AccountNo** field is empty unless the competitor was derived from a related **Contact1** table record. The **Details/Links** tab could contain **Details**, not contained within the **ContSupp** table or **Links** which again are not contained within the **ContSupp** table. This information is stored in yet another table, the **OpMgrFld** table. I have to ask myself, why weren't **Details/Links** stored with all of the other **Details/Links** in the **ContSupp** table?

Note

*This GUI, and tables are used for the **Project Manager** as well as the **Opportunity Manager**. You may see references to the **Project Manager** throughout this chapter.*

Note

I would like to apologize for the quality of some of these screen-shots. I had to exceed the "do not reduce more than 30%" rule in order to fit the screenshot into this book format.

Note

What was our point in this mini explanation?

It was to show you that, though the GUI is a great presentation interface, that it would be virtually impossible to present this information to your boss in a printed report. Especially if you were trying to use the GoldMine reporting tool or even the GoldMine Dashboards unless you were very proficient with either.

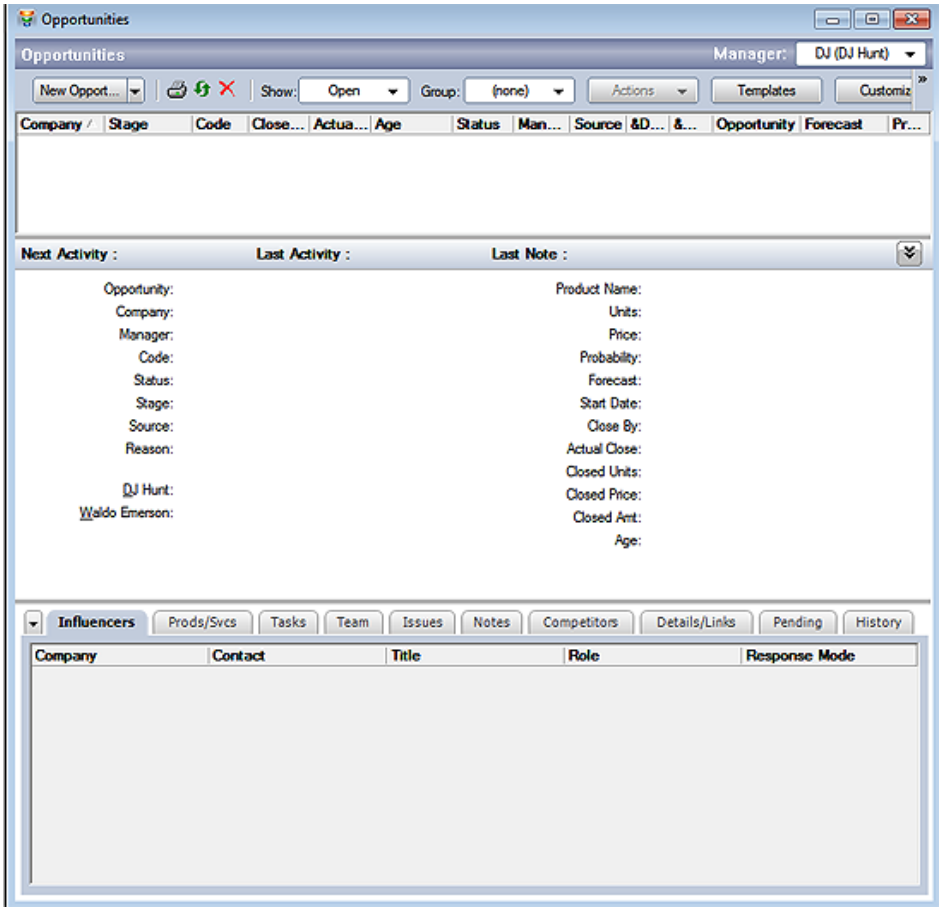


Figure 16-1

Finally, you have the **Pending** and **History** tabs. Both tabs display **Cal** and **ContHist** records, respectively, that are linked to the opportunity.

You can see that there are many tables, and in some tables many one-to-many relationships. This inhibits the user from reporting this information in a paper format. It can be done with an outside reporting tool like Crystal Reports if you have someone that knows how to create a Crystal Report for you. There are outside services (not us) that have specialist to create these Crystal Reports for you. As well there are some built in GoldMine Premium Opportunity/Project Reports as well as the **Opportunity Dashboard** under the **Sales** category in the GoldMine Premium Dashboards. If, however, you cannot make use of these, and you must have paper output, then, possibly, the Opportunity/Project Manager is not the correct tool for you to be utilizing.

While examining Figure 16-1, you can easily see that you can look at the opportunities of any single user, all users, or any user group. The Opportunity Manager defaults to the GoldMine logged in UserID. You may drill down further by using the **Show:** option on this screen. This option allows you to limit your view to those opportunities that are **Open**, **Won**, **Lost**, **Closed**, **Abandoned**, **Postponed**, **Active**, **Completed** (**Won**, **Lost**, **Closed**), and of course **(all)**. As I stated earlier, this is a great GUI for consolidating your opportunity information for display purposes.

Configure the Opportunity / Project Manager

Let's look at some of the customizations that one can use in the Opportunity/Project Manager. The **Customize** button, to the right in the Opportunity Manager Toolbar is the button of choice for this procedure. Clicking this button will display the dialog form shown on the next page in Figure 16-2.

As the Figure 16-2 dialog form shows, you have a **Fields** tab, a **Labels** tab, a **Tabs** tab, and, lastly, an **Options** tab. Here on the **Fields** tab GoldMine is allowing the end user to add up to five new user-defined fields that may be employed in the **Opportunity Properties** dialog form on the **Other** tab of Opportunity dialog form.

As our former Corporate Edition users will remember, you can create a **New** field, **Edit** an existing field, or **Delete** an existing field while on the **Fields** tab. Additionally, you may move a field up in position, or down in position. This has no cause or effect other than in this dialog form. When the **Other** tab (not depicted in Figure 16-2) is created for the **Opportunity Properties** dialog form, the user-defined fields will be listed one on top of each other in alpha order using the GoldMine field name regardless of what you supply for a **Local Label** later on.

Clicking on the **New** button produces the same dialog form as I had previously described and depicted back in Figure 4-2 of Chapter 4, **User Defined Fields**. Your field name must follow the field naming conventions, and may be a **Character**, **Numeric**, or **Date** based field type with the characteristics of the field type that you have defined.

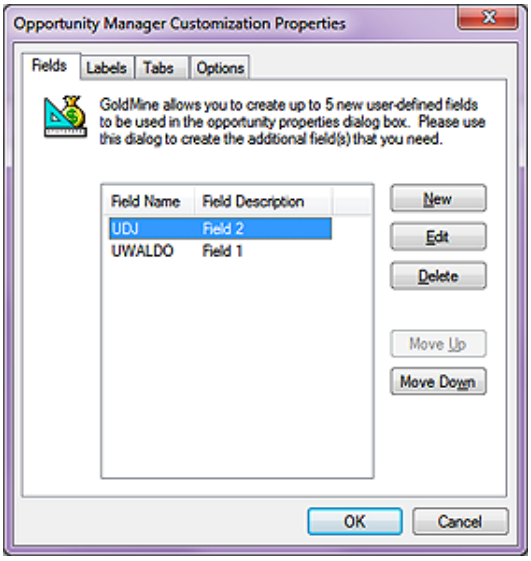


Figure 16-2

Clicking on the **Labels** tab, Figure 16-3, allows you to add or modify any of the labels for the existing opportunity fields, as well as any that you may have created. You do this by double-clicking to the left of the **Default Label** in the **New Label** column for which you wish to change the label.

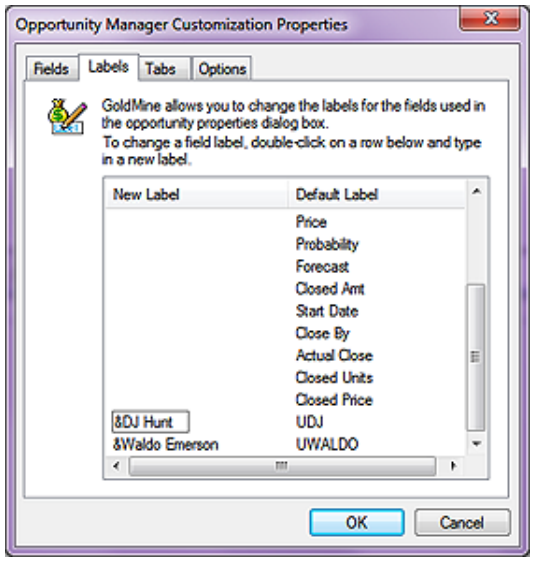


Figure 16-3

This is the place to change the GoldMine defined labels, as well as adding your user-defined field labels (see Sidebar Note). To work with this dialog form you simply double-click in the proper area under the New Label column. You may change any of the default labels, and, at this point, you may add any labels for your user defined fields.

One thing that I should mention is that only about 13.5 characters of your user defined field label will be displayed. I do not advise your exceeding this 13 character limitation for your user defined field labels although you certainly could if you so choose.

In Figure 16-4, I show you the **Tabs** tab. If you are following along in your GoldMine, please click on the **Tabs** tab. It is from here that you can select which of the GoldMine Opportunity tabs, as displayed in Figure 16-1, will actually be displayed, and which order these tabs will appear in for your users to view while in the Opportunity Manager. Remember that I listed the tabs earlier as they stand in their default configuration order.

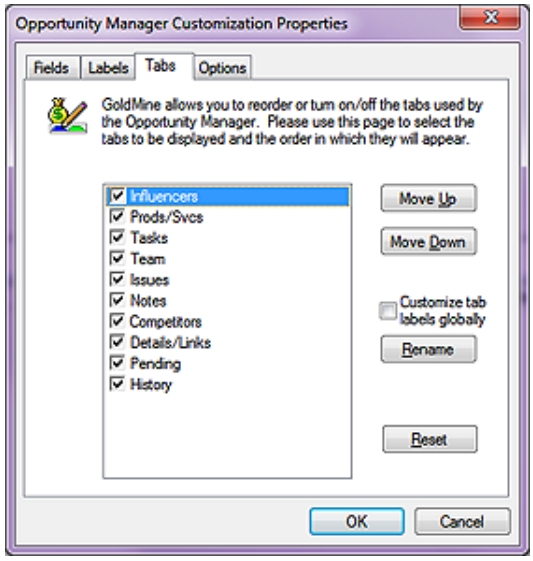


Figure 16-4

It is a very simple procedure. To have the tabs displayed, the boxes should be checked. An unchecked tab box will cause the tab to not be displayed.

To move a tab, first highlight the tab, and then click the **Move Up** or **Move Down** button as is appropriate to your design. It took me weeks of dedicated training, and hard studying before I could figure out how to do this correctly. I'll bet that you can figure it out in 2 seconds.

Since the last printing of **The Hacker's Guide to GoldMine Premium**, the parent from which this book was derived, the FrontRange Developers have seen fit to add a couple more capabilities to this dialog form page. One may now ☐ **Customize tab labels globally**, or **Rename** a tab. For instance, I could have renamed the ☑ **Competitors** tab by highlighting the tab name in the list, and selecting the **Rename**

Note

*I have intensionally shown you the **New Label** edit area to the far left of the **Default Label**. This is where you must click or double-click to edit the new label for the field.*

Note

*Programmers will know this already, but any letter preceded by an ampersand (**&**) will be designated that letter as the **Hot Key** for that label. You must be careful as the hot keys must be unique per screen. Hot keys are honored for the user defined fields as well in theory, however, in this build, I could not test this out in practice. If you have users that prefer to use their keyboard over their mouse, you may want to deploy with hot keys defined.*

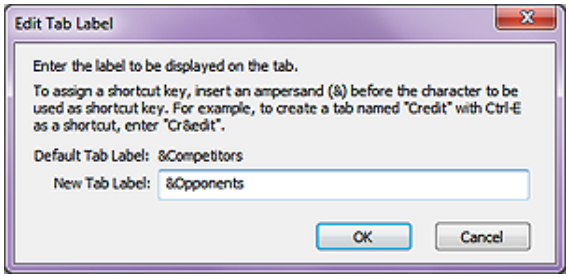


Figure 16-5

button to produce the **Edit Tab Label** dialog form as shown here in Figure 16-5. The **Edit Tab Label** dialog form represents the **Default Tab Label:** in an unmodifiable format, along with the **New Tab Label**: field where you may enter the new name which you wish to give this tab. As I have shown in Figure 16-5, you may also indicate a Hot Key as long as you keep it unique to the **Opportunities** dialog screen.

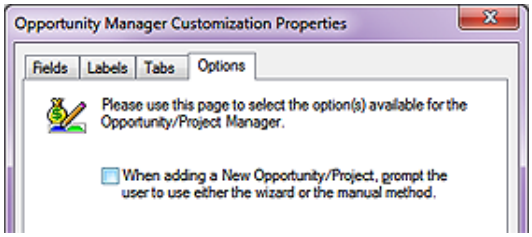


Figure 16-6

I display the **Options** tab here in part in Figure 16-6 only because it exists. Although the tab name is **Options**, plural, on this build, there is but one option, the ☐ **When adding a New Opportunity/ Project, prompt the user to use either the wizard or the manual method.** option. You won't believe this, but this option, when selected, does exactly that which is stated.

At any point in the process, on any tab in the dialog form, if you click on the **OK** button, your changes will be saved, and the dialog form will close.

Opportunity Templates

Before I actually begin my discussion of the GoldMine Opportunity Wizard, I want to discuss Opportunity Templates. Templates tend to help reduce the typing, read that as tend to reduce the typos, while increasing the efficiency, and consistency of the data in your organization. Believe me, that translates to a good thing when you are talking about databases, and their administration. If you have a finite number of products, or services, you may want to create a template for the opportunity to assist your users when they create a new opportunity. With the Opportunities screen open, you may see a **New Opport...** drop list (at least that's how it appears on my GoldMine as the far left button), please click on the drop list arrow and select:

New Template

...from the drop list. Notice, while you have the drop list open, that you could also **Clone Template**, assuming that you had a template to clone. Clicking on the **New Template** drop item will display the dialog form shown below in Figure 16-7.

Note

*The **Template Name:** field validation method (code) is set to not allow a blank in this field. You will not be able to move on to the other field properties until you have established a name for this template.*

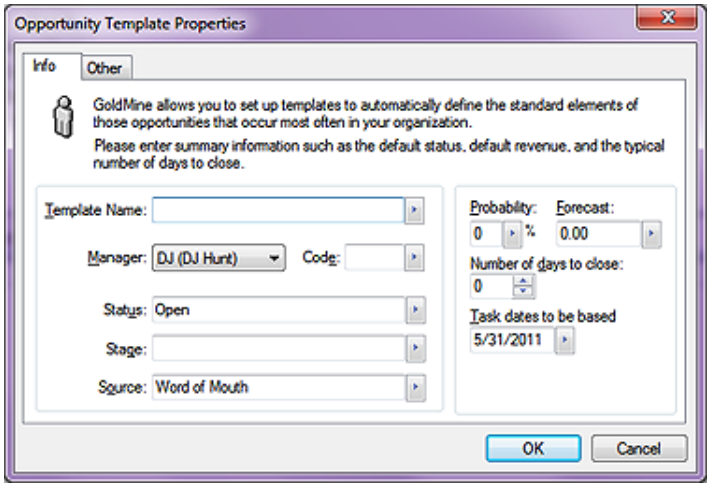


Figure 16-7

This is the **Opportunity Template Properties** dialog form. You may have noticed that there is an **Other** tab on the template form, hence you could also predefine your user defined fields as well, however, not before you supply a **Template Name:**. You may pick from a list of predefined F2 Lookup names, but you probably won't find a good template name there. For this exercise let's just give this one a name like: **Definitive Guide Opportunity Template**.

I would mention here, that there are two frames containing property fields, although the frames are not named frames, they are clearly defined by their respective framing box. I will start by discussing the left most frame.

The first thing that I would want you to do is to give your template a Template Name:, and even though there is an F2 Lookup List associated with the field, I recommend not using it. Templates should have a unique name, and even though GoldMine allows you to add templates of the same

name, how would an end user know which template to utilize, for an Opportunity/Project, if the names were identical. They wouldn't, hence, the need to keep the template names unique, yet meaningful.

The **Manager:** field defaults to the logged in **UserID**, however, you may select any **UserID**, or **User Group** from the drop list for this field.

Should you decide that you are going to have similar template names (against my recommendation), then the **Code:** field could help you differentiate between the various templates in that group when displaying the templates in the top portion of the Opportunity Manager GUI. When creating an opportunity based on a template, only the template name will be displayed. No user will be able to differentiate between the templates based on the name, hence, the users will probably not use the templates, defeating the corporate standards that you are trying to attain.

You will next want to establish a default **Status:** for a new opportunity of this type, and that truly depends on how your organization works an opportunity. The default values in the F2 Lookup List are:

Abandoned - possible for a new opportunity, but not advisable for a template
Closed - possible for a new opportunity, but not advisable for a template
Lost - possible for a new opportunity, but not advisable for a template
Open - very good choice for a new opportunity, and default selection for a template
Postponed - possible for a new opportunity, but not advisable for a template
Won - possible for a new opportunity, but not advisable for a template

I think what I am saying above is that there is really only one possible choice, unless you create your own choices on the F2 Lookup List. Remember, however, before you make any decision to change this lookup list, that the **Status:** display list, **Show:** on the **Opportunities** screen, will only recognize specific status types. Consequently, you could be defeating the functionality of the GoldMine Opportunity Manager should you decide to add or remove items to the F2 Lookup List of the **Status:** field.

The next field, in order, is the **Stage:** field. As with the **Status:** field, after you have determined your corporate settings, I do recommend that you lock down the list (see sidebar Tip). I do not recommend that you allow free form typing into this field. I like the supplied F2 Lookup List, however, should you desire to change it, I recommend that you only change the text, and not the leading numeric values.

The default list is:

10 - Initial Contact
20 - Interest Confirmed
30 - Evaluation Requested
40 - Evaluation INP (In Progress ?)
50 - Evaluation Positive
60 - Evaluation Passed
70 - Making Decision
80 - Verbal Agreement
90 - Purchase Order

This is pretty much self explanatory, however, you may want to change the verbiage to better suit your corporate stages with respect to an opportunity. Again, once established, I do recommend locking down the list, and forcing only a valid input. As a template, you should first determine at which **Stage:** is a client most frequently when the creator of the opportunity would actually create the opportunity using this template. More frequently than not, I will default my templates to **30 - Evaluation Requested**.

The next field of interest is the **Source:** field. I want to mention that this field is the same field as that which is shown in the default GoldMine main contact screen. The lookup list is also the same list as is used in the **Source:** field F2 Lookup list. Additionally, if the underlying record has **Source:** field information, it will be brought into this template by default, and you must change it for your Template definition. I am going to give this a little more thought, however, as of this writing, it is my opinion that this list should be locked down, however, I do not recommend that you employee the force valid input option, and that you do allow blanks. I reserve the right to change my position on this later in the chapter.

Next, we move on to the right side frame, and the first field that we encounter is the **Probability:** field. Though most think that this should be directly tied to the **Stage:** field input, GoldMine does not have the capability for relational lookup lists. At least that is what I wrote in my last book, **The Hacker's Guide to GoldMine Premium**. We now have the capability of referential lookup list. Click on the **Setup** button when in the F2 Lookup List for the **Probability:** field. In the resulting dialog form, be sure to check ☑ **Lookup list depends on another field:**, and then select the radio button ⦿ **Opportunity/Project (OPMGR) field**. From the **Field Name:** list select **Stage**, and click on the **OK** button. Next you'll need to match the **Field Entry** to the **STAGE** value which by default is **(any)**. If you do this properly, then once the Stage: is selected, only the related value will be available from

Note

*I always, yes always, recommend the use of the **Code:** field to my clients. This helps later when doing SQL queries, and other research work.*

*Additionally, this **Code:** field is not limited to the 3 characters, as are the other known code fields. This code field can contain up to **5** characters.*

Tip

*Once you have determined the corporate standard for your template, I recommend that you change the settings of the **F2 Lookup List** to:*

☐ ***Allow blank input***
☑ ***Force valid input***

and to not:

☐ ***Allow adding***
☐ ***Allow editing***
☐ ***Allow deleting***

the **Probability:** F2 Lookup List. Reducing the amount of user input error. Also, you could set the list, and lock it down (as described in previous sidebar Tip).

The default values for this field are:

00 //Lost Interest
20 //Internet Unconfirmed
30 //Confirmed Interest
40 //Selected for Eval
50 //Evaluation INP (has anyone figured out what this acronym, INP, extrapolates out to?)
60 //Evaluation Good
70 //Passed Evaluation
80 //Final Approval INP
90 //Purchase Order INP
99 //Purchased waiting on check

I recommend that you match the template field value here to the template field value that you entered into the **Stage:** field. You could even annotate the value with the matching Stage verbiage, and now that you have the **Probability:** F2 Lookup List link to the **Stage:** value, it should be a lot easier to acquire consistency between the two.

The next field is the **Forecast:** field which is disabled in the template builder. Although there is nothing in the F2 Lookup List by default, if you have some standard opportunity default potentials, I do recommend that you populate the list with those when creating an Opportunity as you can't access the F2 Lookup List here.

The **Number of days to close:** would be the average number of days, from the start to finish, for this type of opportunity, in your organization, to close. Don't worry, it is not a hard, and fast setting. Take your best guess if you don't have the historical figures to back you up at this time. Default this value to 30 days if you are unsure.

Lastly, **Task dates to be based**. Don't do anything here. GoldMine, when using the template, simply picks up the system date of the workstation for adjustment by the creator of the opportunity.

Next, if you have any user defined fields, you may click on the **Other** tab, Figure 16-7, and fill in those fields for the template. At this time, you should establish an F2 Lookup List, and requirements for same. Do this while you have the opportunity (not intended as a PUN).

Click the **OK** button, and save that template.

Note

If these acronyms are as ambiguous to you as they are to me, then I would recommend that you change the verbiage to be more inline with your Corporate Campaigns. I don't, however, recommend that you change the leading numeric values.

The Wizard

Figure 16-8

There are many ways to create a New Opportunity utilizing this GUI, however, the obvious way would be to click on the drop list button which defaults, at least in my GoldMine, to **New Opport...**. If you have not, as yet, turned off the Wizard, this action will start the **Opportunity Wizard:** at the **Welcome** page as shown here in Figure 16-8.

In this first page of the wizard there is little for one to do. By default, GoldMine selects the information **Company:** and **Contact:** from the active contact window. If this is not the intended company, the user may search their database by clicking upon the icon to the right of the **Company:** field. You will notice that the **Company:** field is disabled for typing (however, it is not grayed out), and this field can only be changed by a search through the Contact Search Center for the appropriate company.

Additionally, the **Contact:** field is not editible. There is a drop list of contact names which are related to the Company record. This includes the Primary Contact as well as all listed Secondary or Additional Contacts. In other words, the Opportunity Contact must be a member of the Company record before launching the Opportunity Wizard, you cannot introduce a contact here once the Wizard is in play. As always, you could add the contact after you have finished with the Wizard, and then edit the Opportunity to reflect the new contact. I would, however, refer you to the sidebar Note.

Click on the **Next >** button to advance to the next page of the Wizard.

Note

*Even though the Search icon is to the right of the **Company:** field in Figure 16-8, the **Contact Search Center** is brought up in a **Contact** sort order. Some individuals may find this confusing.*

Note

*If you use the paradigm that I always espouse, **One record for each Contact linked via the Relationship Tree**, then you will only need the Primary Contact here. You can link others to the Opportunity as **Influencers** later on in the Wizard.*

The Wizard now steps us into the **Opportunity Wizard: Description** dialog form page. The user must **Enter or select a descriptive name for the opportunity below:** before they can proceed to the next field or the next page in the dialog form. This is a required field, and you will not be allowed to proceed if you attempt to leave this field empty.

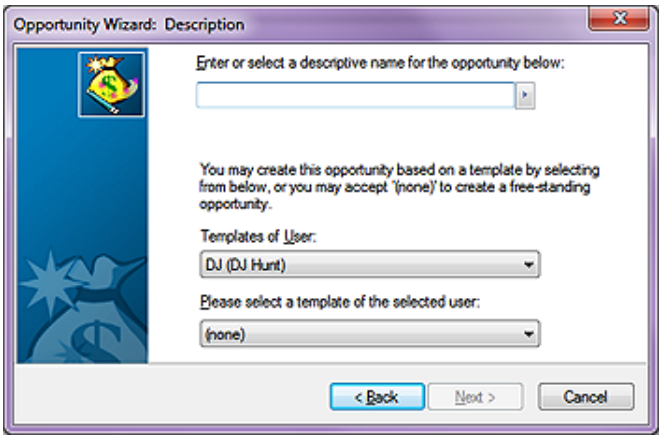


Figure 16-9

As the information statement says: *"You may create this opportunity based on a template by selecting from below, or you may accept '(none)' to create a free-standing opportunity."* I discussed the value of templates earlier in this chapter under the **Opportunity Templates** section. This is a simple matter of first selecting the **UserID** or **User Group** in the **Templates of User:** field, and then selecting a template belonging to that **UserID** in the **Please select a template of the selected user:** field. Again, both are drop lists, and are uneditible, therefore, all of this information must have been predefined.

For this exercise I have entered **GoldMine Premium - The Definitive Guide** as my description, however if you have a lot of repetitively used Opportunities, you may want to consider add those to the F2 Lookup List. Again, for this exercise, let's not use our template so that we can reap the full benefit of the Wizard.

Click on the **Next >** button to proceed.

We next come upon the **Opportunity Wizard: Overview** dialog form which is somewhat pre populated for you. The **Manger:** field defaults to the **UserID** of the logged in GoldMine user, while the rest of the fields are populated as per your template or with default values when not utilizing templates. Alternatively, you would need to adjust the fields appropriately for your needs. If, however, you have standard opportunities, and most will, then I do recommend that you employ templates. It would be to your advantage.

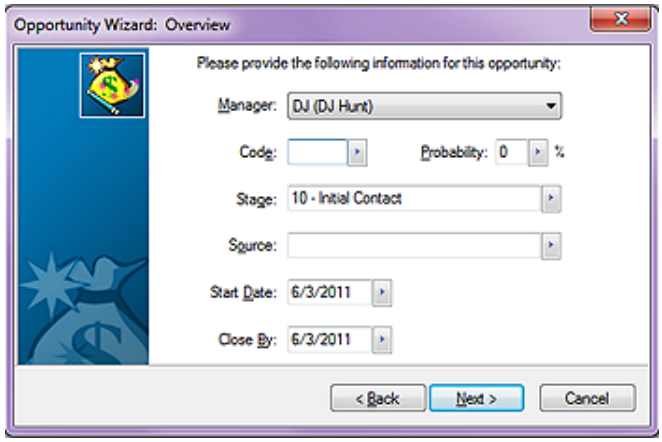


Figure 16-10

Once you have completed this dialog form page of the Wizard, click on the **Next >** button to advance to the next page of the Wizard.

There may be only one person who can influence the outcome of this opportunity, or there may be many. This dialog form page, **Opportunity Wizard: Influencers**, Figure 16-11, allows you to add as many influencers as is necessary. By default, GoldMine assumes that the originally selected Primary Contact, as shown in Figure

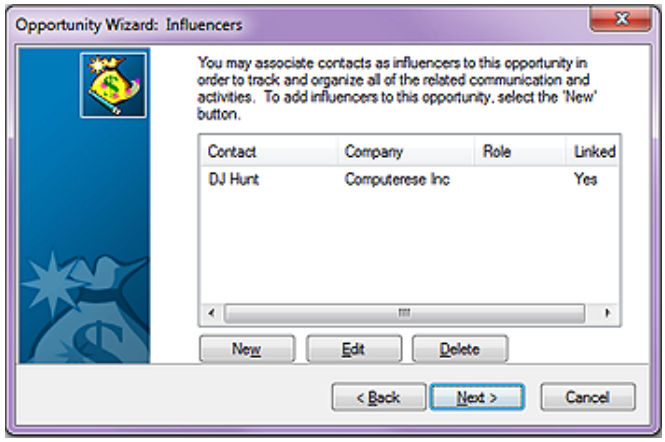


Figure 16-11

Note

*This is a one-to-many relationship, and each record is stored in the **OpMgr** table employing a **RecType** for each of the different grouping of records. This particular grouping, the **Influencers**, uses a **RecType** of **OC**.*

16-8, is an Influencer. You may add as many influencers as you desire, and they do not need to all be from the same contact record, or from any contact record at all for that matter.

If you click the **New** button, you will be required to search or supply the **Company:**, **Contact:**, **Title:**, **Role:**, **Response Mode:**, and/or add any **Notes:** that may be appropriate.

If you click on the **Edit** button, you will be allowed to add or change any of the above mentioned fields for the selected influencers record. You will notice, by default, the primary influencer does not have a **Role:**, **Response Mode:** or **Notes:**.

Click on the **Next >** button to proceed.

Note

*This is a one-to-many relationship, and each record is stored in the **Cal** table employing a **RecType** of **S**. The Forecast Sale has a link to the Opportunity in the **OpMgr** table stored in the **LOpRecID** field. This is the **RecID** of the Opportunity in the **OpMgr** table. The primary record in the **OpMgr** table, as indicated by **RecType** of **O** will be updated each time a linked sale is modified.*

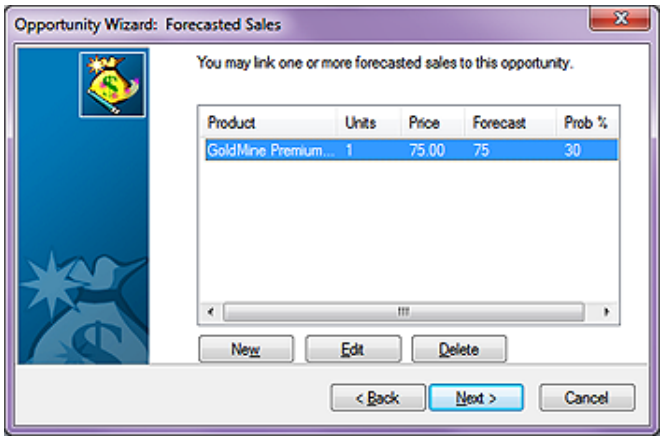


Figure 16-12

This screen, the **Opportunity Wizard: Forecasted Sales** Figure 16-12, is normally empty, and it is up to the creator/modifier of the Opportunity to create a **New** Forecasted Sale. After all, what is an Opportunity, but a Forecast Sale. Some of your templates may default populate this in the new sale dialog form like **User:**, **Probability:**, **Code:**, and **Sale Date:** while the remaining fields will need to be completed by the enduser. This is nothing more than the scheduling of a normal GoldMine Schedule a Forecasted Sale activity that is related to an Opportunity.

You may decide to have only one Forecast Sale per Opportunity, or you may have many Sales, if using the add-on product **MultiCast Gold** for instance. The information in each Forecast Sale is linked to the Opportunity (see sidebar Note), and, as any Sale is edited, the Opportunity Forecast will be updated automatically.

Click on the **Next >** button to proceed.

Note

*This is a one-to-many relationship, and each record is stored in the **OpMgr** table employing a **RecType** for each of the different grouping of records. This grouping, the **Team Members**, uses a **RecType** of **OT**.*

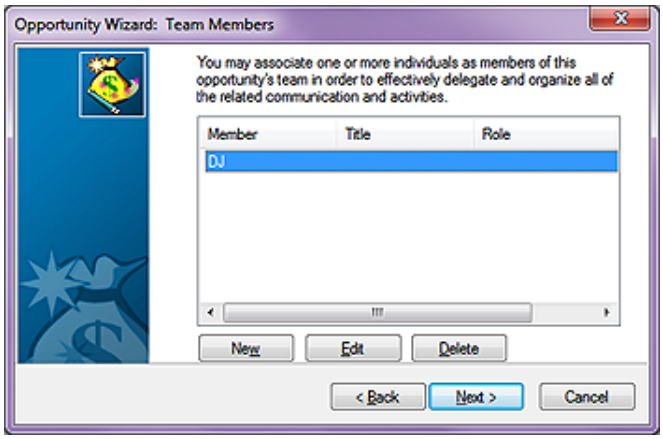


Figure 16-13

On the **Opportunity Wizard: Team Members** dialog form screen, Figure 16-13, you can add as many team members as you desire. They may be based on your UserIDs, or from the Contact records contained within your database. Clicking on the **New** button will present you with a dialog form for **Sales Team Member**, and you will need to select the **Type:**, either **User** or **Other Contact**. If you select **User**, then the **Member:**, **Title:**, and **Department:** fields will be filled in using the Users table, and the User_Var variables. If you select Other Contact, these fields will be automatically completed with the appropriate information from the contacts record. The other fields that you will want to complete are the team members **Role:** and any appropriate **Notes:**. As before, you may add as many or as few team members as will be assisting you in the closing of this Opportunity.

Click on the **Next >** button to proceed.

We now come to the **Opportunity Wizard: Issues** dialog form, Figure 16-14 as displayed at the top of the next page. Each opportunity may have as few as zero Issues or as many Issues that stand in the way of this opportunitys closing. Here you will want to add any issues that present an obstacle to your opportunities closing. When you click on the **New** button, you will be asked to, *"Please en-*

Note

*This is a one-to-many relationship, and each record is stored in the **OpMgr** table employing a **RecType** for each of the different grouping of records. This grouping, the **Issues**, uses a **RecType** of **OI**.*

ter information regarding the key issues related to this opportunity to help you better understand the customer's needs." You should fill in the fields concerning **Issue:**, **Status:**, **Priority:**, **Date:**, the **User:** to which the issue is assigned, and any **Notes:** that may be appropriate. The **Issue:** should be a one liner as to the pertinent issue, while the **Status:** may be **Open** or **Closed** depending on the state of the Issue. **Priority:** is set to not be empty, and you may choose something like **Critical**, **Essential**, **Non-Critical**, or whatever your Administrator has set for the F2 Lookup value list options.

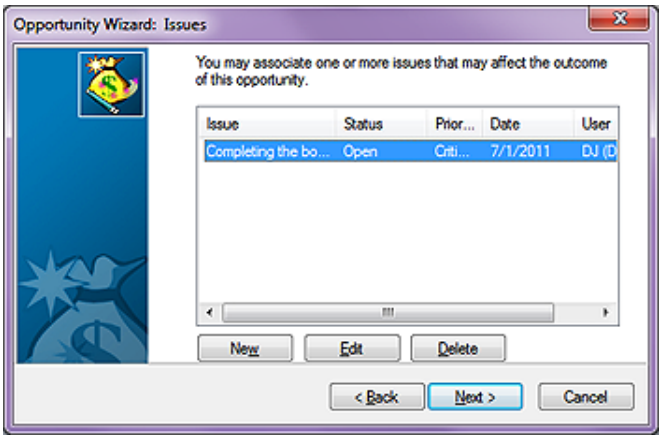


Figure 16-14

Click on the **Next >** button to proceed.

Note

*This is a one-to-many relationship, and each record is stored in the **OpMgr** table employing a **RecType** for each of the different grouping of records. This grouping, the **Competitors**, uses a **RecType** of **OP**.*

As with any opportunity, you may want to keep track of any known competitors, hence the **Opportunity Wizard: Competitors** dialog form, refer to Figure 16-15. Once you click on the **New** button, you may input information free form, or you may select a competitor from one of your existing Contact records. I always keep my competitors right there within my database.

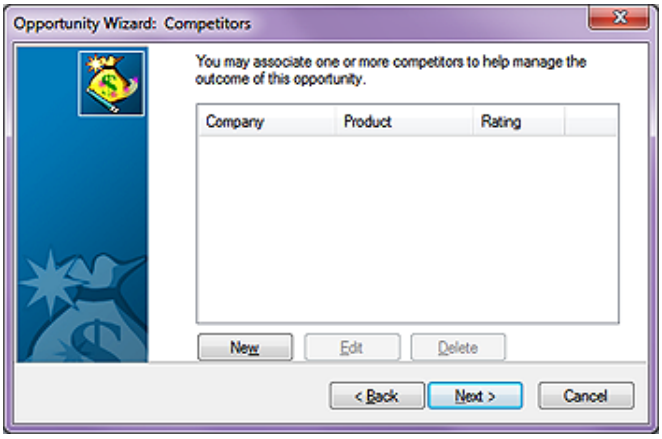


Figure 16-15

Most likely, you will have a standard set of competitors, and you should have a record for each. The information that will be stored in the **OpMgr** table for this **RecType** is: **Competitor:**, **Contact:**, **Rating:**, **Status:**, **Product:**, **Strengths:**, **Weaknesses:**, and, of course, **Notes:**. None of these fields is forced, though there are some default F2 Lookup Lists that you may want to explore. You would want to add any, and all known competitors at this time, although you may add more later as they enter into the Opportunity for those that are price shoppers.

Click on the **Next >** button to proceed.

Note

*This is a one-to-many relationship, and each record is stored in the **OpMgrFld** table employing a **RecType** for each of the different grouping of records. This grouping, the **Details/Links**, uses a **RecType** of **F**.*

Next, we come to a rather confusing wizard dialog form, the **Opportunity Wizard: Details/Links** dialog form, Figure 16-16. Do not think of this in terms of the normal GoldMine Detail in the Contact record display under the tab Details, as it is not. It is for any **Linked Document**, but not in the normal GoldMine sense of the word. You will not be able to view this link under the **Links** tab for instance. It is only viewable via the Opportunity Manager. You must give an **Item:** type (usually selected from the related F2 Lookup List), and a **Reference:**. As well, you may select that this ☐ **Document is GoldMine Merge Form**, and you

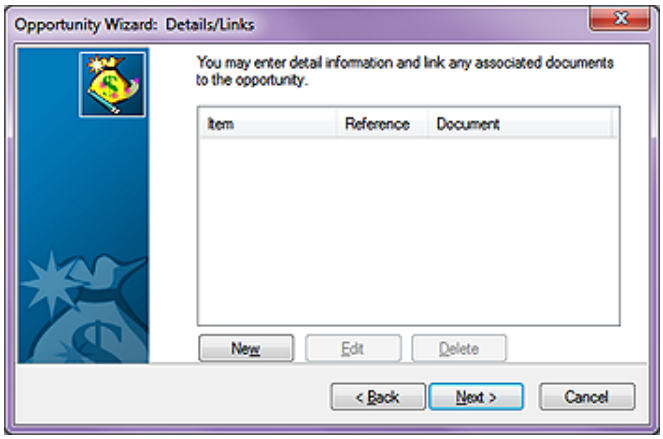


Figure 16-16

may select from your available templates, or you may just browse for a document in the **File Name:** field. You must remember, in this case, that you probably do want to leave the ☑ **Allow File to Synchronize** option selected. That's about all there is to this dialog form.

Click on the **Next >** button to proceed.

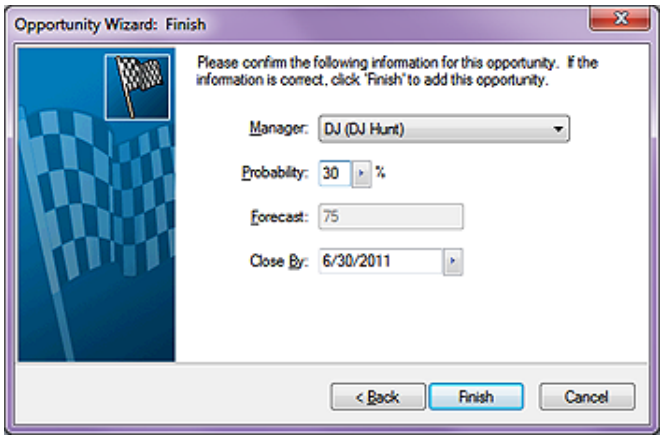


Figure 16-17

And, finally a reconfirmation of some of your previously posted information on the **Opportunity Wizard: Finish** dialog form. You are again asked to confirm the **Manger:**, **Probability:**, and **Close By:** date. You will, as shown in Figure 16-17, be able to see the **Forecast:**, however, it is a **Read Only** field.

Click on the **Finish** button to create your new opportunity.

The Opportunity will, from here on, be displayed, and managed from within the **Opportunities** display form as shown below in Figure 16-18. Users may create, and associate any activity to an existing opportunity via the **Opportunity/Project:** field on the activity creation dialog form. If an activity was not associated at the time of creation, or there was no scheduled activity, an opportunity may be associated upon completing an activity via the **Opportunity/Project:** field. You will notice that these fields contain drop lists, hence, the Opportunity must exist before an activity, Pending or History, can be associated with (related to) an Opportunity.

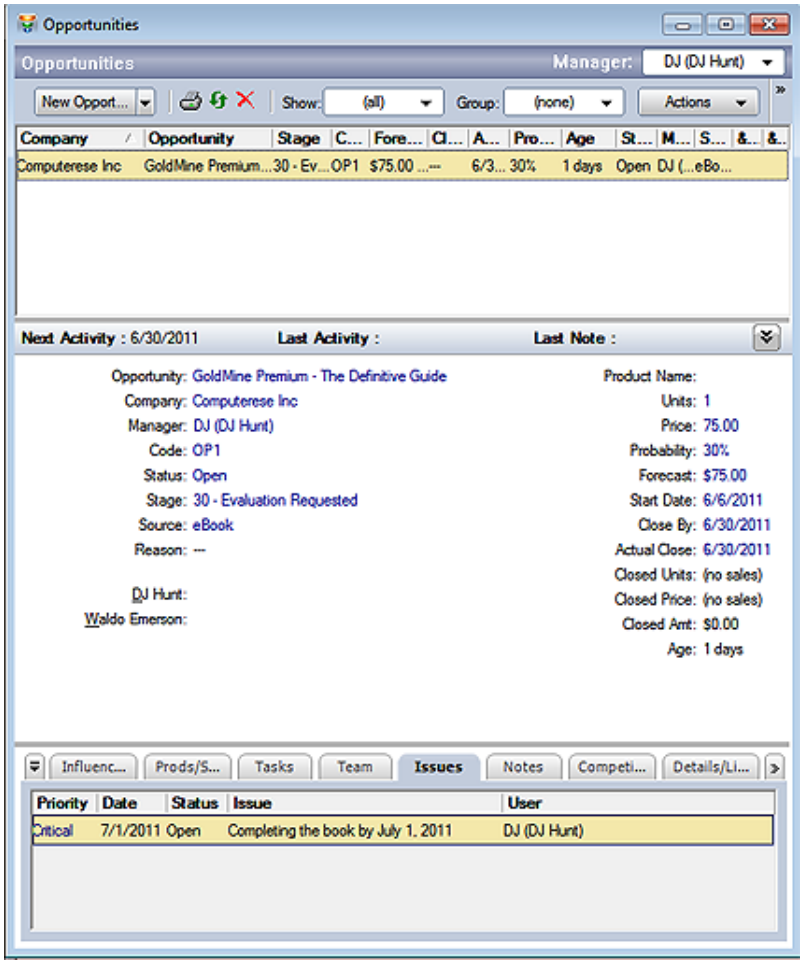


Figure 16-18

This concludes this chapter on The Opportunity Manager.

Note

*I would like to reiterate that opportunity information is best viewed, and worked with via the **Opportunities** screen. If you must have reports on paper, then you may want to forego the use of the **Opportunities** all together.*

17

The Service Center

In This Chapter

About the Service Center

First and foremost, let me make it perfectly clear that the Service Center module in GoldMine Premium is no Heat equivalent. The Service Center was added to the GoldMine Premium product, in my opinion, to have a new feature for the marketing team to utilize in their push to get everyone to upgrade to GoldMine Premium before FrontRange discontinues the rest of the GoldMine product line. As we all know, GoldMine 6.7 and prior have already been formally discontinued, and as you may or may not know, so too has the GoldMine 7 new license sales. As of this writing, FrontRange is still permitting upgrade sales to GoldMine 7, however, this writer expects even that to cease soon.

Additionally, FrontRange is attempting to pull the GoldMine main focus away from Sales by giving it a feature that is not at all Sales related. The Service Center does make GoldMine more of a corporate CRM solution, then strictly a Sales CRM solution as people had tended to pigeon hole the GoldMine product in days of yore.

So let's take a look at what FrontRange is saying about the Service Center. From the GoldMine Premium Help file:

Case management functionality in GoldMine® Premium Edition provides customer service agents the ability to capture service requests, then filter and access the details of each case. The customer service center enables your entire service division to pursue, arrange and filter all service requests and optimize work flow throughout the day.

GoldMine Premium Edition case management features allow your service professionals to assign, escalate and resolve customer service requests quickly and efficiently. Quickly access customer service issues, route them to the service agent who can best solve the problem, and keep clients informed of their service request progress, all within your customized GoldMine environment.

Your GoldMine users can now accept incoming customer service requests from multiple sources, be alerted to upcoming and urgent cases, and get details for any open or closed service request. Individual agents and customer service managers can filter their daily case activities, improving work efficiency and enabling optimal customer care.

The **Service Center**, which we just learned is also frequently referred to as **Case Management** functionality, can be accessed easily from the Outlook style toolbar to the lower left of the GoldMine Premium main screen. Simply click upon the **Service** button. As always, you could access the Service Center from the GoldMine menu:

GoTo
Service Center

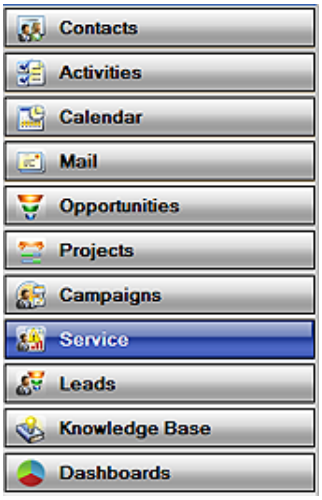


Figure 17-1

Figure 17-2, next page, shows you the result of either action, the **Service Center** dialog form. This dialog form, and the Service module in total, utilizes the **Cases** tables that I had previously discussed in **The Tables** chapter of this book. You may want to familiarize yourself with that chapter, and those tables, again before continuing on with this chapter.

Note

*Unlike previous editions of GoldMine Premium, the **Cutomize** button is no longer hidden and exposed with a Case creation. Although I have squeezed the image to fit the page, I think that you see part of the **Customize** button to the far right in Figure 17-2 along the button bar.*

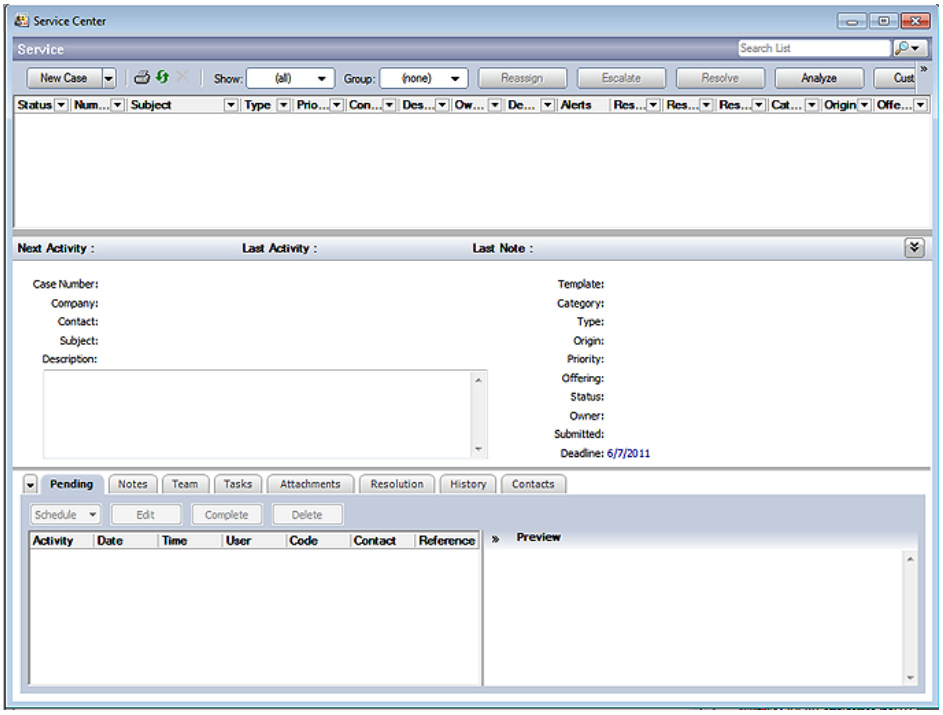


Figure 17-2

Templates

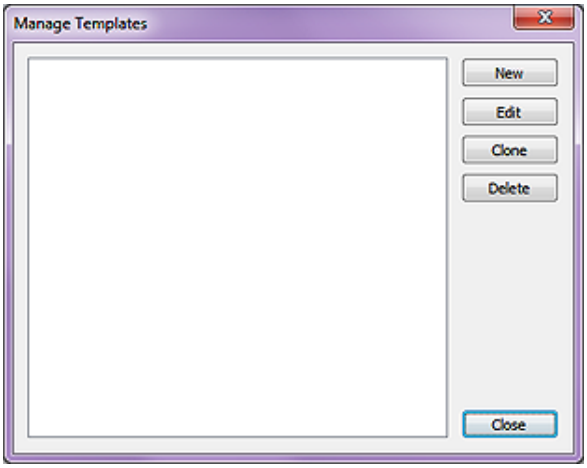


Figure 17-3

Why don't we begin by creating a template. Simply right-click anywhere in the top grid area of the **Service Center** to bring up the Local Menu, and select **Manage Templates** from there. This brings up the **Manage Templates** dialog form as shown here in Figure 17-3.

You will notice that we have the standard buttons of **New**, **Edit**, **Clone**, **Delete** & **Close**, and not a one of them has a hot key associated with it. I am only interested in the **New** button for this writing as the other actions are, by now, clearly understood by you or at least they should be. However, clicking on the **New** button will produce the dialog form known as **Edit Template**, and is displayed for you below in Figure 17-4.

Tip

*Should you decide not to utilize one or two of the fields, then I would suggest that you enter a value of **Do Not Use** into the template for that field.*

*You see, the field will be displayed when the **Case** is displayed. If you give any user an empty field, they will enter something into it, relevant or not, corporate standard or not.*

Note

*As of this writing, the **Product/Service:** field will allow you to enter an unlimited number of characters, however, when you **Save** the template, only **20** of your characters will be saved as part of this template.*

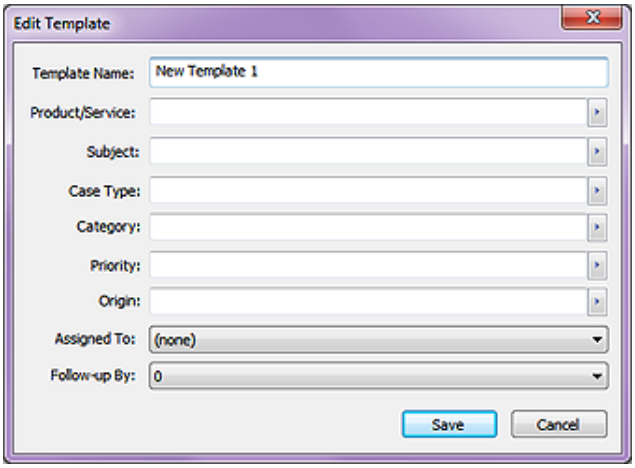


Figure 17-4

Let's briefly talk about what templates you may or may not be wanting to use, and why you may need them. Templates are just that, a starting point for your **Case** record creation. So, do you provide service for one product, or one company where you will probably only need one template? No, most likely not. Most companies will provide service for more than one product and/or more than one company, hence, they will probably have a need for more than one template. You should pre plan your templates with the information that I am about to give you such that when you get to this point in your customization you will be able to create your first run of templates.

For this example, I am only going to create one template, however, I am going to create it as if I were going to be supplying service for multiple GoldMine products. Hence, the first template will be for service for the GoldMine Premium product. Can you figure out what the **Template Name:** will be? Yes, that's right, **GoldMine Premium Issue Ticket**, and as the prefilled in **Template Name: New Template 1** should indicate to you, this is a required field. Hence, if you don't enter a **Template Name:** the default **New Template x** will be entered for you.

In the next field of our template, **Product/Service:**, we will want to enter the product and/or service name for which we will be supplying service support. In this case, I have chosen **GoldMine Premium Edition**. You should notice that the F2 Lookup List for this value is the same list that is used in your **Forecast Sales Products** list. In the **Subject:**, I you are following along, I have chosen to enter **GoldMine Premium Issue**.

Now we come to the **Case Type:** value which also has an F2 Lookup List. I suggest that you think about this, and create your F2 Lookup List at this point. I further suggest that you select from the list, a value be used here. This same suggestion will hold true for any of the other fields that have an F2 Lookup List. So what might a **Case Type:** be? I'm not really sure as I always get this confused and intermingled with the **Category:** field value. Therefore, I'm going to talk about them as if they were one and the same, and it will be up to you to decide as to how you, specifically, wish to use them.

To me, I might consider a **Case Type:** to be **Sales**, **Support**, etc. Of course, I might think the same thing of the **Category:** field. Maybe we should consider using the **Category** field as a placeholder for the **Billable** versus **Non Billable** service. Yes, I'm kind of liking that option so that is what I'm going to enter. For my template value for my **Case Type:**, I am going to enter **Support**, and for the template value for the **Category:**, I'm going to enter **Billable** while putting **Billable** and **Non Billable** on the F2 Lookup List. You can work this around whatever best fits your corporate needs. The best way to determine what has to go in to the fields will be to determine first, what you want out of the Service Center information in your reports, and/or your SQL queries. Always use the output as a guideline for your input.

The next field of our Template will be the **Priority:** field. Again, I suggest that you create and utilize your F2 Lookup List. In this list individuals tend to use word association for priorities. Some examples of this might be:

Cold	**Bronze**	**Red**
Warm	**Gold**	**Yellow**
Hot	**Platinum**	**Green**

Anything that works for your organization is great. For my template I'm setting my **Priority:** field to **Green**.

And now we come to the **Origin:** field of our template. I have set my F2 Lookup list to **E-mail**, **Postal**, **Telephone**, **WebImport** & **Word of Mouth**, and I have selected **Telephone** as the default value for this template.

The next field that we can populate for our template is the **Assigned To:** field. For this field I will usually leave the default value of **(none)**, and this way the **Owner** is automatically assigned the creators **UserID**.

Finally, we encounter the **Follow-up By:** field of our template, and this is nothing but a drop list of days which are automatically applied to the origination date for the default **Deadline:** date of the issue. I have chosen seven days by the way. So our final **Edit Template** dialog form would be as shown here in Figure 17-5. Let's click on the **Save** button to save this as our template. You could create more templates at this time, however, for this exercise I am going to proceed right on to the customizations so go ahead and close down the **Manage Templates** dialog form by clicking on the **Close** button.

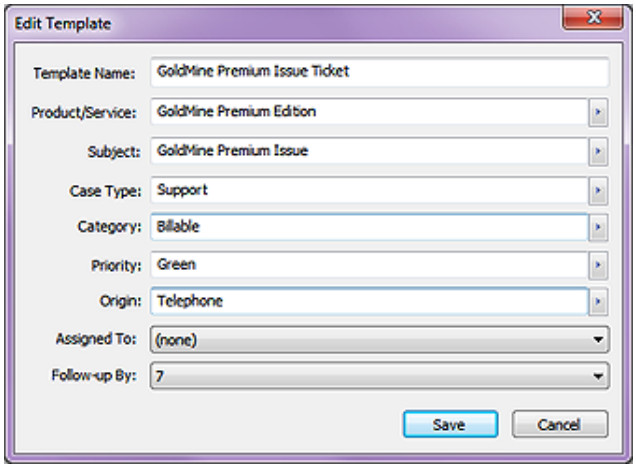


Figure 17-5

Note

*As the **Product/Service:** utilizes the same F2 Lookup List of your **Forecast Sales Products**, your Support Team could utilize this one template for any item in your **Forecast Sales Product** F2 Lookup List.*

Populate that list carefully, and it'll be useful in many locations within GoldMine Premium.

Customize

Note

My company, Computerese Inc, does Support for many other GoldMine Dealers Clients on their behalf. To differentiate these different support tasks, we frequently will change this mask so that the ***Case Number Suffix*** *will reflect the organization to whom the* ***Case*** *belongs.*

Changing the ***Suffix*** *mid-stream will not affect previously created* ***Case Numbers****.*

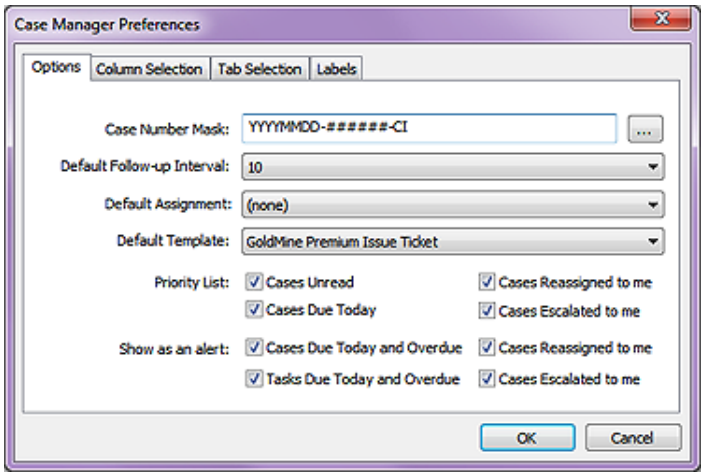


Figure 17-6a

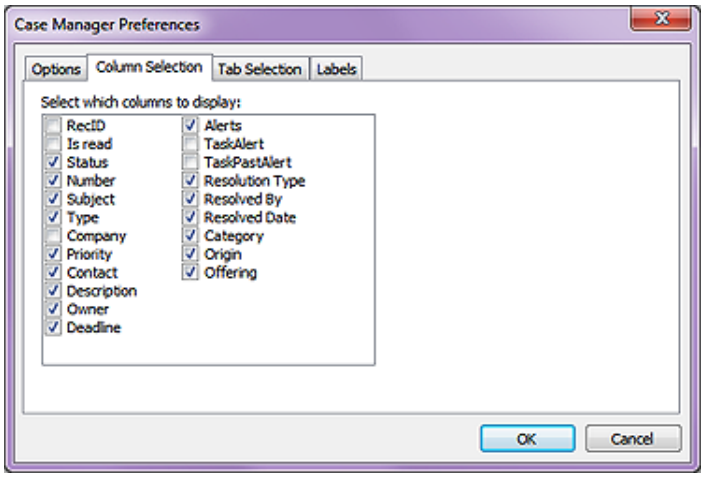


Figure 17-6b

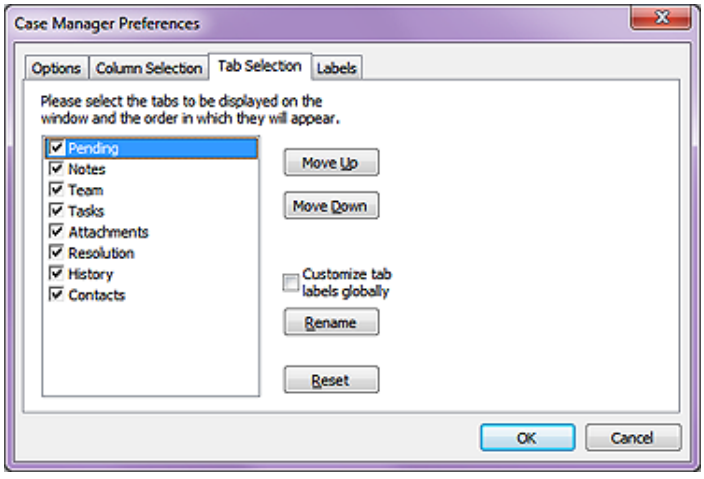


Figure 17-6c

Why don't you next click on the **Customize** button to produce the dialog forms similar to those shown here and on the next page in Figures 17-6a, 6b, 6c & 6d.

I have already done some minor customization under my **Options** tab, hence, these figures may not appear exactly the same as yours will appear. For instance, the **Case Number Mask:** is a derived number mask. Click on the ellipses button to the right of this field to produce the **Set Case Number Mask** dialog form as shown in Figure 17-7 on the next page. This dialog form has a finite set of possible solutions so it is customizable within those constraints.

From the **Set Case Number Mask** dialog form, you can derive a case number mask that you wish to utilized within your company. The **Preview:** field is a non-editable field that will display your mask real-time as you change the **Settings** in the frame below the **Preview:** display.

So let's look and see how that mask is derived by taking a look at the **Settings** frame in Figure 17-7 or on your system. The first field that we encounter is the **Prefix:** field, and even though it says that it is the prefix, it is not if one checks the first ☑ **Use Timestamp** option. You will notice that I have left this field empty, but the choice is yours. Notice to the right of this field that there is an option selection to ☑ **Use Timestamp**. Should you choose this, a timestamp in the designated format, will be utilized as the prefix in lieu of any value that you may have entered into the **Prefix:** field, and not in addition to your entered value. Watch the **Preview:** area to see the results of each action as you tab out of each field.

The next field that we have is the first of two **Separator:** fields from which you may only select predefined values to separate the prefix from the identity. As you can see I have selected to use the hypen (**-**) as the first separator.

As my **Identity Seed:**, I have chosen to accept the default number of **100**. This is a counter value, and this seed is the value at which the counter will begin its incrementing. I don't expect to have

more than 999,999 cases in my organization, hence, I have set the accompanying **Identity Format:** to the number **6**. Most organizations will probably want to boost the number to a much higher value, but that is a corporate decision that will have to be made at the time of the customization. You can always change this mask property at a later date.

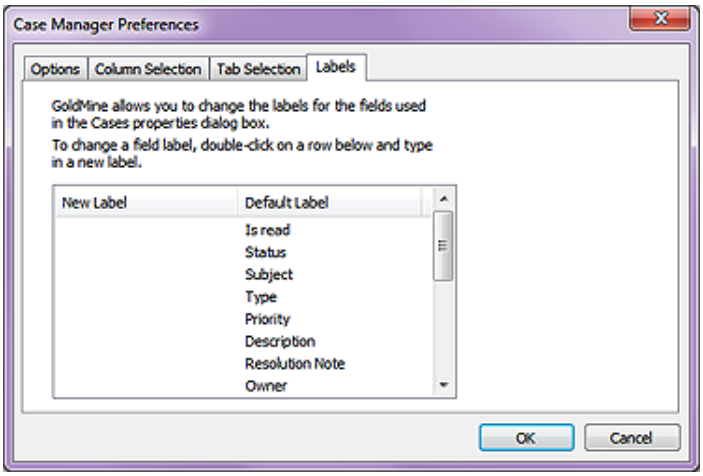


Figure 17-6d

For my **Suffix:** field, I have decided to use **CI** (**C**omputerese **I**nc) along with the hypen (**-**) **Separator:** which is used to separate the **Identity** from the **Suffix** this time.

I liked this mask as displayed in the **Preview:** field, and so I simply click on the **OK** button to accept this mask.

I'm sorry to have you shuffle back and forth like this, but let's finish our discussion of the customizations as depicted in Figure 17-6a on the previous page. The next property is the **Default Follow-up Interval:** in days. This, again, is a drop list, and you may only select a value from the predefined list. In my example, I arbitrarily chose **10** days.

The **Default Assignment:**, as shown in my example, is set to **(none)**. I decided to select **(none)** so that a newly created **Case** will be assigned to the creator of said **Case**.

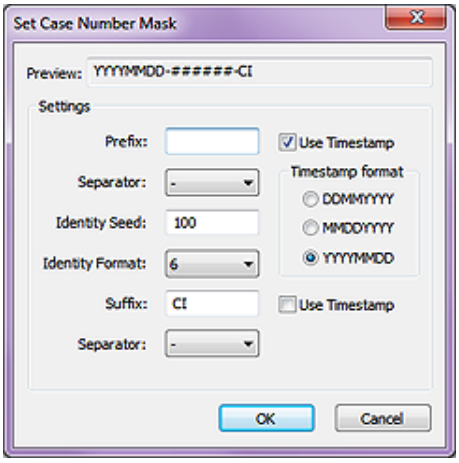


Figure 17-7

You can see that I selected my previously created **GoldMine Premium Issue Ticket** as my **Default Template:**. Hence, the logic behind my creating the template prior to configuring my customizations. See, sometimes I know what I'm doing.

And, lastly, we have a bunch of checkbox options that are all selected in the default state of the **Options** tab of the **Case Manager Preferences** dialog form. I list them here for you, and hope that they are self-explanatory:

Priority List:	☑ **Cases Unread**	☑ **Cases Reassigned to me**
	☑ **Cases Due Today**	☑ **Cases Escalated to me**
Show as an alert:	☑ **Cases Due Today and Overdue**	☑ **Cases Reassigned to me**
	☑ **Tasks Due Today and Overdue**	☑ **Cases Escalated to me**

In Figure 17-6b on the previous page, you can see that you may select those columns that you want to appear when you display a list of Cases in the Service Center. Remember my rule of not trying to put 50 pounds of manure into that 5 pound bag. You choose the columns on the **Column Selection** tab to display, but your screen size, and GoldMine layout determine the number and size of usable columns that you can display. Since everyone using GoldMine may have a different screen size, hence resolution, you may want to keep this to a minimal number of essential columns.

Figure 17-6c on the previous page, represents the **Tab Selection** tan, and is where you determine the tabs that would be available in the Service Center when your user is in the record display mode. I don't understand why you would not display all of the tabs, but that is strictly your decision. Perhaps, for instance, that you don't have any Competitors, hence, you may not.

Figure 17-6d above, represents the **Label** tab, and, as the test states: *"GoldMine allows you to change the labels for the fields used in the Cases properties dialog box."*. To change a label, double-click under the **Label Column** in the row for the **Default Label** that you wish to change, and then enter your desired label.

Go ahead. Make your customizations, and save them.

Note

So you're thinking as the Administrator, that you are customizing the Service Center for everyone. Well, yes you are, but your customizations may not be maintained. You see, everyone that has access to the Service Center also has access to the Customize button, even those without Master rights.

A Process

WARNING

*As of this writing, GoldMine Premium 9.0.2.36, one doesn't appear to be able to create cases for **Additional Contacts** in GoldMine Premium unless you first convert the **Addtitional Contact** to a full GoldMine Contact record which supports my philosophy of one record per Contact linked via the **Relationship Tree**.*

Note

*In the default GoldMine configuration, the Service Center highlighted record does not sync the Contact record. When in the list mode of the Service Center, you may want to have your individuals right-click in the list area, and select **Sync to Contact** from the local menu.*

Note

*Although the **Description:** field is a Memo type field, it is not capable of Rich Text or HTML formatting, hence, an SOP for textual formatting using indents and headers will have to suffice.*

Recommendation

*It has always been my recommendation that a limited number of individuals, who are adequately trained in your SOP's, be allowed to enter new information into your **Knowledge Base.** It is important that the information contained in the Knowledge Base is accurate, and entered in a consistent and organized manner.*

*The better your Knowledge Base, the better your **First Level Support** staff. That is a true enough axiom.*

Okay, we don't need that Case any more. You may right-click on it, and select **Delete** from the local menu. So now, are we ready to begin with your first real Case? Absolutely, let's go for it. For practice, put yourself in the position of a Support Representative, and I'll do the same. We'll walk through a process together. This is not the process, but it is a process that you may or may not wish to use in your organization.

A telephone call comes in to your support line:

Step 01: Click on the **New Case** button, and, while doing that, find out who is on the telephone as the **Contact Search Center** has just popped up for you to enter the information.

Step 02: Locate the record for this contact in the **Contact Search Center**. If you are unable to locate the Contact, then try via Company, or Telephone, or any other of the many ways of relating a record to a Case. A New Case is created for you utilizing the previous customizations.

Step 03: If you work for me that is, you immediately go out of the Service Center, and over to that Contacts record. In my organization, you would first need to validate that this Contact had sufficient Support Time on account to allow you to proceed with support or to collect funds if there was a negative account balance.

Step 04: If they do not have Support Time remaining on their Contract, and they do not want to replenish their account, then you politely explain the situation, hang up the telephone, and **Resolve** the Case. For this exercise, however, we will assume that this Contact had sufficient Support Time to proceed with the incident.

Step 05: Validate and change, if necessary, all Case information displayed. Specifically, make sure that you acquire a complete **Description:** of the issue. It is important that your organization establish **S**tandard **O**perating **P**rocedures (**SOP**s) for Technical Support. For instance: If the caller has received an error message try to get the error message exactly as it is displayed on their screen, and place this information into the **Description:** field. You see, this will assist you when you attempt to **Resolve** the issue.

Step 06: Although I won't go into each tab specifically, during the entire support process it is important to take copious notes, and the technician should use the resources available when working on a Case. For instance: you may have noticed the **Pending**, **Notes**, **Team**, **Tasks**, **Attachments**, **Resolution**, **History** & **Contacts** tabs (not necessarily in that order). Use them. If you must research something, or have something researched for you, **Schedule** an **Activity** or a **Task**. If you require assistance to get to the resolution, then add to your **Team**. If you are sending the Case caller any documents, then **Attach** them. I can't emphasize enough how important it is to have historical information in the Support Center when it comes to working with future Cases.

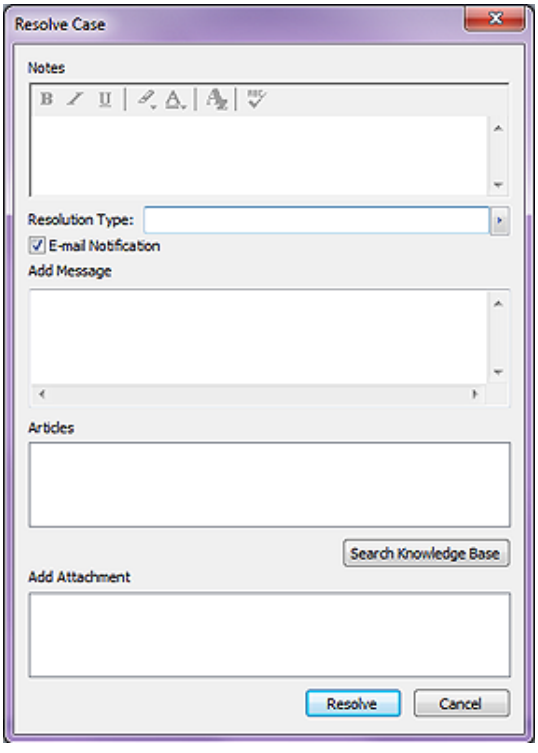


Figure 17-8

Step 07: Click on the **Resolution** tab, and then click on **Search Knowledge Base** button to search for an immediate solution. This is why it is important, if an error message is thrown, that you capture that information. It is certainly more precise to search for **Error 104**, for instance, than **GoldMine not working**. If you don't find a resolution in your **Knowledge Base**, then you would probably want to click the **Add Note** button on this tab to add a comment as to what you have done so far.

Step 08a: If you have Resolved the issue to the callers satisfaction through the use of the Knowledge Base, then you would want to click on the **Resolve** button which will bring up the

Resolve Case dialog form as shown in Figure 17-8 on the previous page. You will notice many fields that must be completed, and it is important that you have a **S**tandard **O**perating **P**rocedure established for the use of this dialog form, and that your Service individuals have been fully educated in your Corporate **S**tandard **O**perating **P**rocedures. Having this **SOP**, with regards to the resolution, will assist the GoldMine Administrator, yes you, in your building of the Knowledge Base. The better your Knowledge Base, the more effective your First Level Support team will be.

So here is how I envision using this dialog form. The **Notes** that you enter here will be displayed under the **Resolution** tab in the **Notes** area of that tab.

The **Resolution Type:** is an F2 Lookup List that is, by default empty, so FrontRange has really not given us any clue as to the purpose of this field value. When you enter a value into this field, and resolve the case, the value is stored in the **Cases.Resolution_Type** field. Yet, I can find no hint of this value associated with the case in the Service Center. Possibly this information is being stored for later Reports or SQL Queries. The 3 options that I have in my F2 Lookup List for the **Resolution Type:** field are:

Abandoned
Client Hung Up
Resolved

Next, in this dialog form, we have a simple checkbox option, ☑ **E-mail Notification**. This option is selected by default, and it's probably an option that you would not want to unselect. This will automatically create an E-mail message for the contact with information about the resolution if the related Contact has an E-mail Address. **S**tandard **O**perating **P**rocedures again should be to capture the proper information before beginning the interview.

In the next memo field, **Add Message**, you are permitted to put information here to be included in the body of the E-mail message that is being sent to the caller. This could be hand typed information or it could be information that you have gathered by cutting and pasting from your Knowledge Base. Whatever this information is, it will be brought over to the body of the E-mail message as Plain Text.

Next, we have what appears to be a memo field, however, you may not type into the **Articles** area. You may search your Knowledge Base, and the title of any article that you select from the Knowledge Base will be entered into this field. This information will not, repeat not, be placed into the E-mail message. Instead, upon resolution, this item will go into the **Notes** section of the **Resolution** tab with a hotlink to the selected document in the Knowledge Base which, as you will see, is totally different than the next section.

In the **Add Attachment** area of this dialog form, if you have any attachments attached to this Case under the **Attachments** tab, all of these attachments will be displayed as unselected checkbox items. In the **Add Attachment** area, you may check those attachments that you wish to have attached to the callers E-mail message, and, upon resolution and creation of the E-mail message, you will see the selected attachments attached to the E-mail. This is quite a different action from the previous section wouldn't you say.

At this point you would simply click on the **Resolve** button to accept the resolution, and to send the E-mail if so selected.

Step 08b: The alternative to the **Resolve** is that the first level technician cannot resolve the issue. As an alternative to resolving the issue the first level technician may **Reassign**, **Escalate** or even **Abandon** this support call. For this step, we will assume that this first level technician cannot resolve the issue, and instead of escalating it at this time, that they wish to send it to another first level technician for their analysis. In this case they would select the **Reassign** button which will produce a dialog form similar to that shown here in Figure 17-9.

You will notice that the **Case List** has automatically been populated with the case number that is being reassigned.

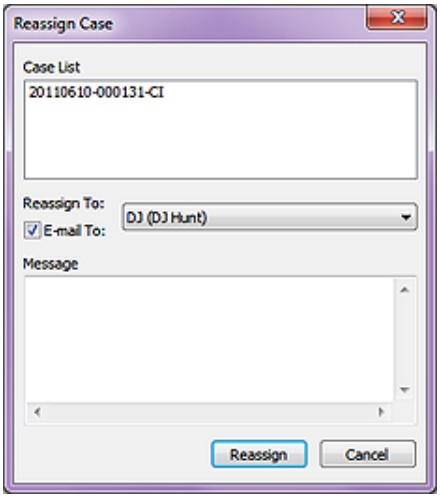


Figure 17-9

Note

When using GoldMines Internal E-mail, you would be wise to have your users turn on the alarms for messages. As stated in Chapter 3, this feature only works for GoldMine Internal Messages. Alternatively, you could do a ***GM.ini*** *override to cover everybody on the network.*

[GMAlarm]
OnByDefault=ACTSOMDL

Note

With respect to User Friendliness, I would have expected all of the ***Resolution*** *tab items to be clustered (framed) together, and all that pertained to the E-mail message to be clustered (framed) together separately.*

The **Reassign To:** option is a drop list from which you may select a GoldMine **UserID**. Additionally, and by default selected, you have the ☑ **E-mail To:**. This will send a E-mail message to the user selected as defined by the Administrator in their **User Settings**. To reassign a case without notifying the person to whom you are a reassigning this case would be an unconscionable act. Hence, you may want to include a small Plain Text **Message** in the body of your E-mail. At this point you would simply click on the **Reassign** button, and wipe your hands of the responsibility for this Case.

Note

As a matter of protocol, I would tend to pass the Case through two first level support technicians before I would consider escalating this to the next level of technicians. This, however, is solely dependent upon your technician structure.

Step 08c: As was previously stated, you always have the alternative option of Escalating this Case to the next level of technicians. There is no difference between the **Reassign Case** dialog form, as shown in Figure 17-9, and the **Escalate Case** dialog form other than the title. Hence, I will not repeat the item by item explanation which is stated above.

Step 08d: At any point in the Case process, it is always possible to **Abandon** the Case. Possibly, the caller has to leave for a meeting after having waited on the telephone for a support technician already for an hour, and they say to *"Forget It"*. Simple enough, you **Abandon** the Case by right-clicking on the Case in question, and selecting **Abandon Case** from the Local Menu. Then complete the **Abandon Case** dialog form as shown her in Figure 17-10.

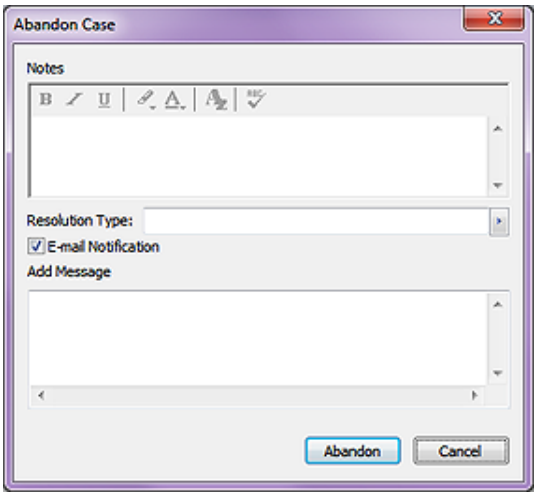


Figure 17-10

This means that you will need to enter some **Notes** as to what happened while you were responsible for this Case. As well, you will need to select or type in a **Resolution Type:** which may be nothing more than **Abandoned**. I always recommend that you set up your F2 Lookup List, and Force a valid input from that list. Remember the old **G**arbage **I**n/**G**arbage **O**ut (**GIGO**) principle. Don't let that happen to your database information.

And, lastly, we have the ☑ **E-mail Notification** to the Case Contact. Along with that you may want to include a small message in the E-mail via the **Add Message** Plain Text memo field of this dialog form. At this point you would click on the **Abandon** button, and be done with this Case.

Before leaving the Support Center, I did want to mention, for your Support Team Managers, that there is an **Analyze** button available that will produce some valuable information about your team efforts. You may do an analysis against any UserID, User Group or even for everyone in GoldMine in a given date range. Once you have entered your parameters, it is a simple matter of clicking upon the **Refresh** button to perform the analysis on the **Case Management Analysis** screen. You may **Export** the resulting information to Microsoft Word or Excel, or you may go directly to the printer with the **Print...** button.

Knowledge Base

So you think that we are at the end of The Service Center chapter do you? If that is what you thought then you would be wrong. You see, earlier I stated that your first level support team is only as good as your **Knowledge Base**, and I wanted to expound on that just a bit.

Most organizations will not dedicate their technicians to a Support Center, not at the price they are paying their technicians any way. Most will hire people people to man their Support Center, and who know absolutely nothing about the product for which they are rendering support. Sure, a conscientious person might review your product manuals, yet, for the most part, they must rely on the information within the Knowledge Base. Hence, the importance of a corporate standard for developing and maintaining your Knowledge Base on a regular basis, and one or two individuals dedicated to this task and this task alone. Well, at least in the first few weeks of implementation.

You can see an example that I created for you in Figure 17-11 on the next page. Remember this is only one option for configuration of many possible options, but make use of the Rich Text formatting capabilities for greater readability. The only thing that really matters is that you standardize on a good format, and that you force everyone to adhere to that standard. This is why it is better to limit the number of individuals actually involved in populating your organizations Knowledge Base or at least have a team of editors review what has been added new each week.

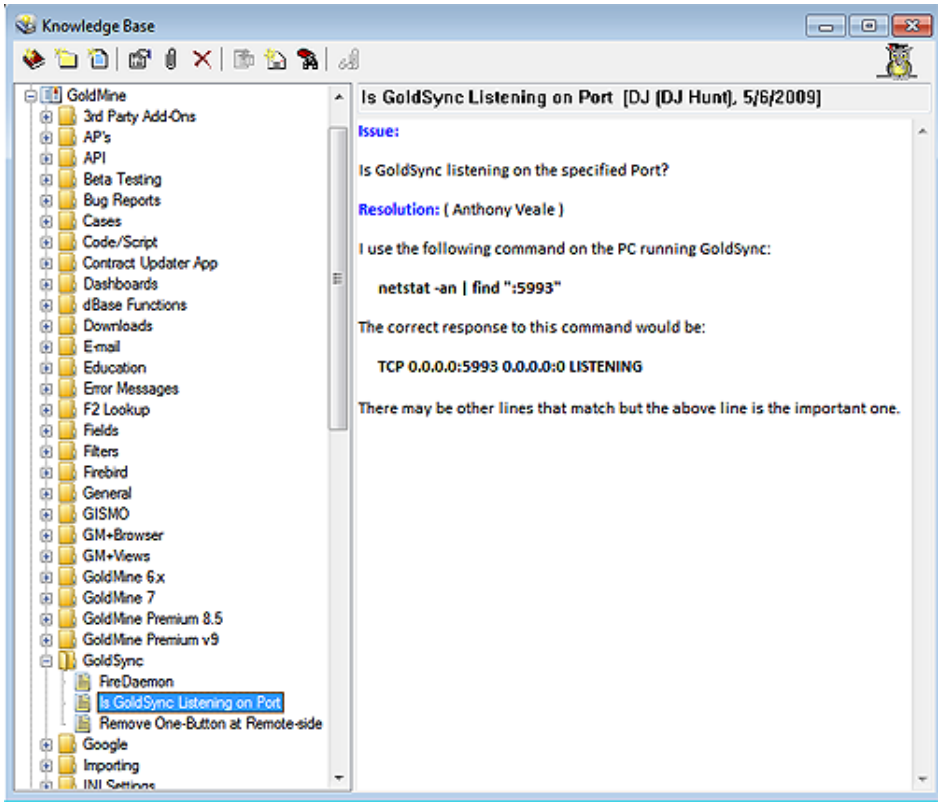


Figure 17-11

Either way, I want to conclude this chapter by reiterating that: Your first level support team can only be as good as is your Knowledge Base. It is incumbent upon you to hire the most personable people that you can for your first level support team, and that you have a well developed Knowledge Base for them to employ in their daily activity. The Service Center is only as good as your corporate standards, and the enforcement of the adherence to those standards.

A

In This Chapter

These are most of the dBase functions that may be used in various areas within the GoldMine Premium application. As you may or may not remember, the dBase functions are still employed throughout GoldMine Premium even when using the Microsoft SQL backend. In fact I recently asked this question directly to FrontRange:

"dBase functions in Filters, screens etc. will not be changing, correct?"

This is the answer that was returned:

"That's the intent. We are not completely sure if we will be able to compensate for all areas, but we do intend to minimize the impact."

Initial testing with this the GoldMine Premium version 9.0.2.36 shows this statement to be true. Actually, I don't see how they could replace dBase which a simpler language for the end users use.

Given that, I have tried to lay these functions out showing the syntax and some examples of usage. Where possible, I have identified locations where the function may or may not be employed. I supplied some functions that we have been able to use, but that are not necessarily supported by GoldMine.

Appendix A

Character Functions

Syntax: **at(<Search for Character>, <Search in Character String>)**
rat(<Search for Character>, <Search in Character String>)

Abstract: The **at()** syntax was designed to allow users to find a character set within a character value starting from the left hand side of the character value and looking toward the right. The **at()** syntax returns a numeric value, representing the first occurrence of that character set from the left hand side, that can be used in other syntaxes (see the **substr()** syntax). The **at()** function should be thought of as **lat()** (not a valid expression) or search the string left to right stopping at the first occurance of, while the **rat()** function is just the opposite, or search the string right to left stopping at the first occurance of. The first argument is the character set for which you are looking. The second argument is the text string, or field, that you want to have searched for the character set.

Prerequisite: At what position does the period occur in the string "Donald J. Hunt"?

Example: **at(".","Donald J. Hunt")** **rat(".","Donald J. Hunt")**

Returned: **9** as a numeric value **9** as a numeric value

Prerequisite: Contact1.Contact = "Donald J. Hunt "

Example: **at(" ", Contact1.Contact)** **rat(" ", trim(Contact1.Contact))**

Returned: **7** as a numeric value **10** as a numeric value

Syntax: **alltrim(<Character String>)**

Abstract: The **all trim()** syntax will remove any leading and any trailing spaces from a character string. Also see, the **ltrim()** syntax and the **trim()** syntax.

Example: **alltrim(" DJH ")**

Returned: **DJH** as a character value

Syntax: **char(<Extract from String>, <Position>) (GoldMine Reports)**

Abstract: The **char**acter syntax takes two arguments. The first argument is the string from which you want to extract the character. The second argument is the position number of the character that you want to extract. Also review the **left()** syntax, the **right()** syntax, and the **substr()** syntax.

Example: **char("The quick brown fox jumped over the log", 5)**

Returned: **q** as a character value

Prerequisite: Contact2.Comments = "The quick brown fox jumped over the log"

Example: **char(Contact2.Comments, 11)**

Returned: **b** as a character value

Syntax: **ctod(<Character Based Date>)**

Abstract: The **c**haracter **to d**ate**()** syntax will convert any character-based date to a real date that can then be evaluated against another date value. This syntax can only be used when the argument being passed is a character type in the form of **"mm/dd/yyyy"**.

Prerequisite: Contact1.Key2 = "6/14/2011 "

Example: **ctod(trim(Contact1.Key2))** **ctod("06/14/2011")**

Returned: **6/14/2011** as a date value **6/14/2011** as a date value

Syntax: **left(<Extract From String>, <Length>)**

Abstract: The **left()** syntax takes two arguments. The first argument is the string from which you want to extract the characters. The second argument is the total number of characters that you want to extract from the left side of the string given in the first argument. Also review the **mid()** syntax, the **right()** syntax, and the **substr()** syntax.

Prerequisite: Contact1.Contact = "Donald J. Hunt "

Example: **left("Donald J. Hunt", 6)** **left(Contact1.Contact, 9)**

Returned: **Donald** as a character value **Donald** J. as a character value

Syntax: **len(<Character String>)**

Abstract: The **len**gth**()** syntax will return a numeric value representing the number of characters in the string that is given as an argument.

Prerequisite: Contact1.Contact = "Donald J. Hunt "

Example: **len("Donald J. Hunt")** **len(trim(Contact1.Contact))**

Returned: **14** as a numeric value **14** as a numeric value

Syntax: **lower(<Character String>)**

Abstract: The **lower()** syntax will convert a character string to all lower case letters. This syntax is often used when comparing two strings that may be dissimilar by first equalizing both sides of the equation. Donald J. Hunt does not equal DONALD J. HUNT, but if both are converted to lower case then donald j. hunt will equal donald j. hunt. See the syntax **upper()**.

Example: **lower("DJH")**

Returned: **djh** as a character value

Example: **lower("Donald J. Hunt") = lower("DONALD J. HUNT")**

Returned: **True** as a boolean value

Syntax: **ltrim(<Character String>)**

Abstract: The **l**eft **trim()** syntax will remove any leading spaces from a character string. Also see, the **alltrim()** syntax and the **trim()** syntax.

Example: **ltrim(" DJH ")**

Returned: **"DJH "** (notice that the trailing spaces remain)

Syntax: **ltrimpad(<Character String>, <Length>, <Pad Character>)**

Abstract: The **l**eft **trim pad()** syntax will remove any leading spaces from a character string, and then pad the resulting string with the pad character to make the length of the string equal to the request length.

Example: **ltrimpad(" 1.00", 8, "$")**

Returned: **$$$$1.00** as a character value

Syntax: **mid(<Character String>, <Start>, <Length (optional)>)**

Abstract: The **mid()** syntax takes three arguments. The first argument is the character string from which you want to extract a specified number of characters. The second argument is the starting position from which you want to begin your extraction; everything from that point to the end of the string will be extracted if, the optional, Length argument is not added. The Length parameter describes how many characters from the starting position to extract. Also review the **left()** syntax, the **right()** syntax, and the **substr()** syntax.

Prerequisite: Contact1.Contact = "Donald J. Hunt "

Example: **mid("Donald J. Hunt",11, 4)** **mid(trim(Contact1.Contact), 7, 7)**

Returned: **Hunt** as a character value **J. Hunt** as a character value

Syntax: **pad(<Character String>, <Length>, <Pad Character (optional)>, <Mode>)**

Abstract: The **pad()** syntax gives you a couple of ways to manipulate a character string. The pad character is optional as it will default to a space. There are 3 pad modes:

0 = right pad
1 = center pad
2 = left pad

Example: **pad("207.00", 8, "$", 0)** **pad("207.00", 8, "$", 1)**

Returned: **207.00$$** as a character value **207.00** as a character value

Example: **pad("207.00", 8, "$", 2)**

Returned: **$$207.00** as a character value

Syntax: **padl(<Character String>, <Length>, <Pad Character>)**
padr(<Character String>, <Length>, <Pad Character>)

Abstract: The **pad l**eft**()** or pad right syntaxes gives you a way to make a numeric character string a specified number of characters in length. If the string being passed is not the specified number of characters, then these syntaxes will insert the character, that you designate, to bring it to the specified length.

Example: **padl("207",6,"0")** **padr("207",6,"0")**

Returned: **000207** as a character value **207000** as a character value

Prerequisite: Contact2.uMyNumber = 122 (as a numeric value)

Example: **padl(ltrim(str(Contact2.uMyNumber)),10,"0")**

Returned: **0000000122** as a character value

Syntax: **proper(<Character String>)**

Abstract: The **proper()** syntax will take any character string between quotations or in a field, and convert the first letter of each word to upper case while the remaining letters of that word are converted to lower case.

Examples: **proper("DONALD J. HUNT")** **proper("donald hunt")**

Returned: **Donald J. Hunt** **Donald Hunt**

Prerequisite: Conact1.Key1 = "EMPLOYEE "

Example: **proper(trim(Contact1->Key1))**

Returned: **Employee**

Syntax: **right(<Character String>, <Length>)**

Abstract: The **right()** syntax takes two arguments. The first argument is the string from which you want to have the characters extracted. The second argument is the number of characters, from the right, which you want to have extracted from the string in the first argument. Also review the **left()** syntax, the **mid()** syntax, and the **substr()** syntax.

Prerequisite: Contact1.Contact = "Donald J. Hunt "

Example: **right("Donald J. Hunt", 4)** **right(trim(Contact1.Contact), 4)**

Returned: **Hunt** as a character value **Hunt** as a character value

Syntax: **space(<Number>)**

Abstract: The **space()** syntax may be used to blank out a field via the use of the Lookup.ini, or may be used in an expression to added a fixed amount of spaces.

Example: **space(0)** **"DJ"+space(6)+"20030201"**

Returned: **empty field** as a character value **DJ 20030201** as a character value

Syntax: **stod(<String Date>)**

Abstract: The **s**tring **to d**ate**()** syntax will convert any character-based date to a real date that can then be evaluated against another date value. This syntax can only be used when the argument being passed is a character type in the form of **"yyyymmdd"**.

Prerequisite: Contact1.Key2 = "20110614 "

Example: **stod("20110614")** **stod(trim(Contact1.Key2))**

Returned: **6/14/2011** as a date value **6/14/2011** as a date value

Syntax: **strtran(<Character String>, <Search String>, <Replace String>)**

Abstract: The **str**ing **tran**sposition**()** syntax allows you to convert a portion of a character string to another character string. This syntax requires that three parameters be passed, all of which are character-based. The first parameter is the string which is to be searched. The second parameter is the string to look for in the first parameter, and the third parameter is the string which you want to substitute for the second parameter.

Example: **strtran("20100614", "2010", "2011")**

Returned: **20110614** as a character value

Prerequisite: Contact1.Key2 = "20100614 "

Example: **strtran(Contact1.Key2, "2010", "2011")**

Returned: **20110614** as a character value

Syntax: **substr(<Character String>, <Start>, <Length (optional)>)**

Abstract: The **substr**ing**()** syntax allows you to parse out any portion of a character string. The first argument is the character string to be parsed. The second argument is the numeric value of the position at which to begin the parsing, while the third argument represents the numeric value of how many characters to parse. You could use the **at()** syntax or the **rat()** syntax as the starting position or ending position.

Example: **substr("Donald J. Hunt", 1, 6)**

Returned: **Donald** as a character value

Prerequisite: Contact1.Key1 = "Donald J. Hunt "

Example: **substr(Contact1.Key1, at(" ", Contact1.Key1)+1, 7)**

Returned: **J. Hunt** as a character value

Prerequisite: Contact1.Key1 = "Donald J. Hunt "

Example: **substr(Contact1.Key1, rat(" ", trim(Contact1.Key1))+1, 4)**

Returned: **Hunt** as a character value

Syntax: **text(<Number/Date>) (GoldMine Reports)**

Abstract: The **text()** syntax takes one argument. This argument can be either a date or a number. The text syntax returns a character representation of the date/number.

Example: **text(Contact1.LastDate)**

Returned: **06/14/11** as a character value

Syntax: trim(<Character String>)
rtrim(<Character String>)

Abstract: The **trim()** and **r**ight **trim()** syntaxes will remove any trailing spaces from a character string. Also see, the **alltrim()** syntax and the **ltrim()** syntax.

Example: trim(" DJH ") rtrim(" DJH ")

Returned: " DJH" as a character value " DJH" as a character value

Syntax: upper(<Character String>)

Abstract: The **upper()** syntax will convert a character string to all upper case letters. This syntax is often used when comparing two strings that may be dissimilar by first equalizing both sides of the equation. Donald J. Hunt does not equal donald j. hunt, but if both are converted to upper case, then DONALD J. HUNT will equal DONALD J. HUNT (see the syntax **lower()**).

Example: upper("djh") upper("Donald J. Hunt") = upper("donald j. hunt")

Returned: **DJH** as a character **True** as a boolean value

Syntax: val(<Character String>)

Abstract: The **val**ue**()** syntax will convert a character string number to a numeric value.

Example: val("10.25") val("10.00")

Returned: **10.25** as a numeric value **10** as a numeric value

Example: val("10.95")

Returned: **10.95** as a numeric value

Syntax: word(<Character String>, <Number>) (GoldMine Reports)

Abstract: The **word()** syntax takes two arguments. The first argument is the string from which you want to have the word extracted. The second argument is the number of the word that you want to extract. Also review the **left()** syntax, the **right()** syntax, and the **substr()** syntax.

Example: word("The quick brown fox jumped over the log", 4)

Returned: **fox** as a character value

Prerequisite: Contact2.Comments = "The quick brown fox jumped over the log "

Example: word(Contact2.Comments, 8)

Returned: **log** as a character value

Numeric Functions

Syntax: **abs(<Number>) (GoldMine Reports)**

Abstract: The **abs**olute**()** syntax takes one numeric argument. This syntax converts a number to its absolute value.

Example: **abs(-10.153)** **abs(-15)**

Returned: **10.153** as a numeric value **15** as a numeric value

Syntax: **ceiling(<Number>)**

Abstract: The **ceiling()** syntax returns the nearest integer that is greater than or equal to the numeric value.

Example: **ceiling(5.2)** **ceiling(-4.3)**

Returned: **6** as a numeric value **-4** as a numeric value

Syntax: **floor(<Number>)**

Abstract: The **floor()** syntax returns the nearest integer that is less than or equal to the numeric argument supplied.

Example: **floor(-10.153)** **floor(3.25)**

Returned: **11** as a numeric value **3** as a numeric value

Syntax: **max(<Number1>, <Number2>) (GoldMine Reports)**

Abstract: The **max**imum**()** syntax takes two arguments. The first and second argument must be numeric values, and this syntax returns the larger of the two values. This syntax can be used to compare two numeric fields.

Example: **max(10, 20)**

Returned: **20** as a numeric value

Prerequisite: Contact2.uNumber1 = 20 and Contact2.uNumber2 = 50

Example: **max(Contact2.uNumber1, Contact2.uNumber2)**

Returned: **50** as a numeric value

Syntax: **min(<Number1>, <Number2>) (GoldMine Reports)**

Abstract: The **min**imum**()** syntax takes two arguments. The first and second argument must be numeric values, and this syntax returns the smaller of the two values. This syntax can be used to compare two numeric fields.

Example: **min(10, 20)**

Returned: **10** as a numeric value

Prerequisite: Contact2.uNumber1 = 20 and Contact2.uNumber2 = 50

Example: **min(Contact2->uNumber1, Contact2->uNumber2)**

Returned: **20** as a numeric value

Syntax: **round(<Number>, <Decimal Places>) (GoldMine Reports)**

Abstract: The **round()** syntax takes two numeric arguments. The first argument is the number that you want to round. The second argument is the number of decimal places to which you want to round.

Example: **round(10.153, 2)** **round(10.153, 1)**

Returned: **10.15** as a numeric value **10.2** as a numeric value

Syntax: **str(<Number>, <Length>, <Decimal Places>, <Pad Character>)**

Abstract: The **str**ing**()** function must be passed a numeric argument, and will convert that argument to a character-based string. The first argument is the number to be converted. The second argument is the length of the string to which to convert the number, and the third argument is the number of decimal places to be represented in the returned string. The fourth argument is the pad character which is padded to the right to fill the length request.

Example: **str(132.50, 6, 2)** **str(132.50, 10, 2)**

Returned: **132.50** as a character **" 132.50"** as a character

Example: **str(132.50, 5, 1)** **str(132.50, 10, 2, "$")**

Returned: **132.5** as a character **$$$$132.50** as a character

Example: **str(132.50, 3)**

Returned: **133** as a character (5 and higher rounds up)

Date Functions

Syntax: **accdate(Contact1.AccountNo)**

Abstract: The **acc**ount number **date()** syntax is used to extract the creation date of a contact record from the GoldMine AccountNo field.

Prerequisite: AccountNo = 98110237007'!$6Q.Jam

Example: **accdate(Contact1.AccountNo)**

Returned: **11/2/1998** as a date value

Syntax: **age(<Date>)**

Abstract: The **age()** syntax can be used only when the argument being passed is a date value. The age syntax returns a number value representing the age from the date given, up to today's date.

Example: **age(stod("19481123"))** **age({11/23/1948})**

Returned: **62** as a numeric value **62** as a numeric value

Prerequisite: Contact1.Key2 = 11/09/1958

Example: **age(ctod(Contact1.Key2))**

Returned: **52** as a numeric value

Syntax: **date()**

Abstract: The **date()** syntax returns the current date of your computer system, and it returns it as a date value for comparison against other date values. The date syntax does not accept any arguments.

Example: **date()**

Returned: **6/14/2011** as a date value

Syntax: **day(<Date>)**

Abstract: The **day()** syntax can be used only when the argument being passed is a date value. The day syntax returns a number value representing the day portion of a date value only. If you convert a character to a date first, you may then use that as your argument. Also see date().

Example: **day(date())** **day(ctod("01/09/1998"))**

Returned: **25** as a numeric value **9** as a numeric value

Prerequisite: Contact1.Key2 = 01/09/1998

Example: **day(ctod(Contact1.Key2))**

Returned: **9** as a numeric value

Syntax: **dobindays(<Date>)**

Abstract: The **d**ate **o**f **b**irth **in days()** syntax takes any date argument and returns the number of days remaining to the month/day of the date argument.

Example: **dobindays({11/23/1948})** **dobindays(ctod("01/09/1998"))**

Returned: **162** as a numeric value **162** as a numeric value

Prerequisite: Contact1.Key2 = 01/09/1998

Example: **dobindays(ctod(Contact1.Key2))**

Returned: **324** as a numeric value (return values dependant on processing day)

Syntax: **dow(<Date>)**

Abstract: The **d**ay **o**f **w**eek**()** syntax returns the weekday from a date value as a number, this is a 0 based value.

0 = Sunday
1 = Monday
2 = Tuesday
3 = Wednesday
4 = Thursday
5 = Friday
6 = Saturday

Example: **dow(date())** **dow({01/09/2003})**

Returned: **3** as a numeric value 4 as a numeric value

Syntax: **doy(<Date>)**

Abstract: The **d**ay **o**f **y**ear**()** syntax returns the number of days from the beginning of the year to the argument month/day value.

Example: **doy(date())** **doy({06/23/2003})**

Returned: **49** as a numeric value **173** as a numeric value

Syntax: **dtoc(<Date>)**

Abstract: The **d**ate **to c**haracter**()** syntax can be used only when the argument being passed is a date value. The date to character syntax turns a date value into character value to be used when strings are required. The returned value will always be in the form of mm/dd/yy. Also see date().

Example: **dtoc(date())** **dtoc({01/09/1998})**

Returned: **02/10/03** as a character value **01/09/98** as a character value

Prerequisite: Contact2.uMyDate = {1/9/1998}

Example: **dtoc(Contact2.uMyDate)**

Returned: **01/09/98** as a character value

Syntax: **dtos(<Date>)**

Abstract: The **d**ate **t**o **s**tring**()** syntax can be used only when the argument being passed is a date value. The date to string syntax turns a date value into character value to be used when strings are required. The returned value will always be in the form of **yyyymmdd**. Also see **date()**.

Example:	**dtos(date())**	**dtos({06/15/2011})**
Returned:	**20110615** as a character value	**20110615** as a character

Prerequisite: Contact2.uMyDate = {06/15/2011}

Example: **dtos(Contact2.uMyDate)**

Returned: **20110615** as a character value

Syntax: **fmttime(<Character Time>)**

Abstract: The **f**or**m**a**t** **time()** syntax will convert a 24 hour clock time value to a 12 hour clock time value.

Prerequisite:	Contact1.LastTime = 21:09	Contact1.LastTime = 08:37
Example:	**fmttime(Contact1.LastTime)**	**fmttime (Contact1.LastTime)**
Returned:	**9:09p** as a character value	**8:37a** as a character value

Syntax: **wdate(<Date>, <Format>)**

Abstract: The **w**rite **date()** syntax takes two argument. The first is a standard date argument, while, with the second argument, one tells the syntax how the resulting character value should be formatted. There are 4 formats available. An example of each is supplied.

Example:	**wdate(date(), 0)**	**wdate(date(), 1)**
Returned:	**Jun 15, 11** as a character	**Wed, Jun 15, 11** as a character
Example:	**wdate(date(), 2)**	**wdate(date(), 3)**
Returned:	**Jun 15** as a character	**Wednesday, June 15, 2011**

Syntax: **weekday(<Date>) (GoldMine Reports)**

Abstract: The **weekday()** syntax takes one date argument, and returns the string day of the week for the date argument.

Example:	**weekday(Sys.Date)**	**weekday(Contact1.LastDate)**
Returned:	**Monday** as a character value	**Tuesday** as a character value

Syntax: **year(<Date>)**

Abstract: The **year()** syntax can be used only when the argument being passed is a date value. The year syntax returns a number value representing the year portion of a date value only. If you convert a character to a date first, you may then use that as your argument. Also see **date()**.

Example:	**year(date())**	**year({06/15/2011})**
Returned:	**2011** as a numeric value	**2011** as a numeric value

Prerequisite: Contact1.Key2 = "06/15/2011 "

Example: **year(ctod(Contact1.Key2))**

Returned: **2011** as a numeric value

Miscellaneous Functions

Syntax: **asc(<Character>)**
chr(<ASCII Number>)

Abstract: The **asc**ii**()** syntax returns the ascii number equivalent of the character enclosed in the parenthesis. The **ch**a**r**acter**()** syntax will return the ascii character that is associated with the number in the argument.

Example: **asc("M")** **chr(190)**

Returned: **77** as a numeric value **3/4** as a character value

---

Syntax: **counter(<Character Var>, <Increment Nbr>, <Start Nbr>, <Action Nbr>)**

Abstract: The **counter()** syntax returns a sequence of consecutive numbers, incremented as specified, each time that the syntax is evaluated. The Character Variable must be unique and is stored in the Lookup table as is the last used number. The Start Number is the number at which you wish the counter to begin, and should only be used once or when resetting the counter to a position. When the Action is set to 1 the counter is reset to the Start Number. When the Action is set to 2 the counter is removed from the Lookup table.

Example: **counter("AcctNo", 1)** **counter("AcctNo", 1, 1000, 1)**

Returned: **1** as a numeric value **1000** as a numeric value

---

Syntax: **double(<Any Type>) (GoldMine Reports)**

Abstract: The **double()** syntax takes one argument, a numeric, character, or date value, and converts this to a numeric value.

Example: **double("10.153")** (with report field properties set to 2 decimal places)

Returned: **10.15** as a numeric value

Example: **double(7/4)** (with report field properties set to 2 decimal places)

Returned: **1.75** as a numeric value

Example: **double("6/15/2011")** (with report field properties set to 0 decimal places)

Returned: **20110615** as a numeric value

Syntax: **html2txt(<Character>)**

Abstract: The **html2txt()** syntax takes one character string argument, and formats the string removing all html coding with the appropriate character string in plain text. This syntax is primarily used to convert HTML Calendar/History Notes inf reporting purposes, although it can be used anywhere.

Example: **html2txt(ContHist.Notes)**

Returned: **This is a test for the HTML2Txt syntax** as plain text value

Syntax: **httpstr(<Character>, <Option>)**

Abstract: The **httpstr()** syntax takes one character argument, usually a website, and formats the string replacing all non-letter/non-number characters with appropriate http percent codes.

Example: **httpstr("http://www.DJHunt.US/Beyond Gold.htm", 1)**

Returned: **http%3A%2F%2Fwww.DJHunt.US%2FBeyond%20Gold.htm** as a character value

Example: **httpstr(Contact1.AccountNo, 1)**

Returned: **A1050268152%29%60%2C%3FYRJai** as a character value

Note

*The immediate if function is your most valuable function in your arsenal of functions. It can be utilized in field level **Record Typing**, in your **Lookup.ini**, and many other locations. Make certain that you understand this function fully.*

Syntax: **iif(<Logical Expression>, <True>, <False>)**

Abstract: The **i**mmediate **if()** syntax will evaluate the first argument, which must result in a True or False when evaluated. If the first argument is evaluated as True, then the second argument will be returned to the calling statement. A False evaluation will cause the third argument to be returned to the calling statement.

Prerequisite: Contact1.Key1 = DJH

Example: **iif(Contact1.Key1 = "DJH", "Top Dog", "Low Man")**

Returned: **Top Dog** as a character value

Note

*The immediate if function will only compare the equal size components. Since you only asked for a comparison to **"D"** only the first character of the **Key1** field is looked at, therefore, this is also evaluating to **True**.*

Caveat: iif(Contact1.Key1 = "D", "Top Dog", "Low Man")

Returned: **Top Dog** as a character value

Example: **iif(trim(Contact1.State) = "MA", "Massachusetts", "Bad State")**

Returned: **True** returns Massachusetts
False returns Bad State

Example: **iif(trim(Contact1.State) $ "MA NH VT ME CT", "Y", "N")**

Returned: if the State field contains MA, NH, VT, ME or CT then **Y** any other state value then **N**

Syntax: **int(<Any Type>)**

Abstract: The **int**eger**()** syntax takes one argument, a numeric, character, or date value, and converts this to an integer numeric value.

Example: **int(10.153)** (Works everywhere in GoldMine)

Returned: **10** as a numeric value

Example: **int("123.50")** (Only works in GoldMine Reports)

Returned: **123** as a numeric value

Prerequisite: Contact2.uMyDate = 6/15/2011

Example: **int(Contact2->uMyDate)** (Only works in GoldMine Reports)

Returned: **20110615** as a numeric value

Syntax: **reccount()**

Abstract: The **rec**ord **count()** syntax does not accept any arguments, and will return the number of records contained in your dBase or SQL version of GoldMine as a numeric value. This number will represent all records, including deletions in the dBase version of GoldMine.

Example: **reccount()**

Returned: **16819** as a numeric value

Syntax: **recno()**

Abstract: The **rec**ord **n**(**o**)umber**()** syntax does not accept any arguments, and will return the current table pointer record number as a character value. This syntax will return RecID on SQL databases.

Example: **recno()**

Returned: **121** as a character value

Syntax: **time()**

Abstract: The **time()** syntax returns the current time of your computer system, and it returns it as a date value. The time syntax does not accept any arguments. The time is returned as a 24 hour value in the format of hh:mm:ss

Example:	**time()**	**left(str(time()), 5)**
Returned:	**21:19:40** as a date value	**21:19** as a character value

Syntax: **user2full(<Character UserID>)**

Abstract: The **user 2 full()** syntax will return the full UserID and User Name, as displayed in GoldMine pick lists. The first example shows you the syntax in its natural usage while the second example shows it in combination with other syntaxes to extract the users full name.

Example: **user2full("DJ")**

Returned: **DJ (Donald J. Hunt)** as a character value

Prerequisite: Contact1.Key1 = DJ

Example: **strtran(substr(user2full(Contact1.Key1), at(" ",user2full(Contact1.Key1))+2,255),")"," ")**

Returned: **Donald J. Hunt** as a character value

Syntax: **UserInGrp(<Integer/String Group>)**

Abstract: The user in group syntax will return a True if the currently logged in user of GoldMine is in the passed user group number or the passed user group name.

Example:	**UserInGrp(2)**	**UserInGrp("Sales")**
Returned:	**.T.** as a logical value	**.T**. as a logical value

B

In This Chapter

Macros

This appendix contains most of the known DDE macro functions that GoldMine Premium will let you employ. This appendix will also try to define when and where these macros may be employed if there are special cases. I have included all of the macros of which I was aware of at the time of this writing.

Some of the areas that will be able to employ these macros are, your **Word** document templates, and your **E-mail** templates whether done in GoldMine or in an HTML editor. You may also use macros in **Filters**, **Groups**, **Automated Processes**, and when you are doing a **Global Replace**. There are a group of macros that will only be usable by application designers.

I can't count how many times I have been asked if there is a way to modify these macros, and my answer is always a resounding, No. GoldMine macros may not be modified by the end user in any way.

Appendix B

Macros

Syntax: **&Address**

Abstract: The **address** macro was designed to incorporate the 3 primary address fields into a label format while removing any blank lines.

Prerequisite: Contact1.Address1 = 150 Pratt Road
Contact1.Address2 = Suite A
Contact1.Address3 = Upper Deck

150 Pratt Road

UpperDeck

Example: **&Address** **&Address**

Returned: **150 Pratt Road**
Suite A
Upper Deck

150 Pratt Road
Upper Deck

Syntax: **&Address1**

Abstract: The **address1** macro was designed to pull only the information contained in the Contact1.Address1 field. In most cases, one could simply have used Contact1.Address1 instead of the macro. This macro does trim off any trailing spaces in the field.

Prerequisite: Contact1.Address1 = 150 Pratt Road
Contact1.Address2 = Suite A
Contact1.Address3 = Upper Deck

Example: **&Address1**

Returned: **150 Pratt Road**

Syntax: **&Address2**

Abstract: The **address2** macro was designed to pull only the information contained in the Contact1.Address2 field. In most cases, one could simply have used Contact1.Address2 instead of the macro. This macro does trim off any trailing spaces in the field.

Prerequisite: Contact1.Address1 = 150 Pratt Road
Contact1.Address2 = Suite A
Contact1.Address3 = Upper Deck

Example: **&Address2**

Returned: **Suite A**

Syntax: **&Address3**

Abstract: The **address3** macro was designed to pull only the information contained in the Contact1.Address3 field. In most cases, one could simply have used Contact1.Address3 instead of the macro. This macro does trim off any trailing spaces in the field.

Prerequisite: Contact1.Address1 = 150 Pratt Road
Contact1.Address2 = Suite A
Contact1.Address3 = Upper Deck

Example: **&Address3**

Returned: **Upper Deck**

Note

*Stating that a macro is a **(Programmers Macro)** does not mean to imply that users cannot employ these macros if they need to, but, instead, that these macros are more commonly used by application developers.*

Syntax: **&BrowseRecNo** (Programmers Macro)

Abstract: The **browse rec**ord **n**(**o**)umber macro was designed to send the programmer the record number of the last selected record in the current browse window. If, for example, the Pending tab were active when this macro was executed, the last selected record number on that Pending tab would be returned. Against SQL tables one should expect that the RecID would be returned as there are not record numbers in SQL tables.

Example: **ddeRequest(nChannel, [&BrowseRecNo])**

Returned: **3648** or **987YCSD(FE]VR#Y**

Syntax: **&CalRefresh** (Programmers Macro)

Abstract: The calendar refresh macro was designed to allow the programmer to programmatically refresh the calendar after it had be modified or append to using other dde macros.

Example: **ddeRequest(nChannel, [&CalFresh])**

Returned: **n/a**

Syntax: **&City**

Abstract: The **city** macro was designed to extract the Contact1.City field from the currently active record in the GoldMine session. This macro does trim off any trailing spaces in the field.

Prerequisite: Contact1.City = Fitchburg

Example: **&City**

Returned: **Fitchburg**

Syntax: **&CityStateZip**

Abstract: The **city**, **state**, and **zip** macro was designed to pull together, in a label format, the fields Contact1.City, Contact1.State, and Contact1.Zip

Prerequisite: Contact1.City = Fitchburg
Contact1.State = MA
Contact1.Zip = 01420-4142

Example: **&CityStateZip**

Returned: **Fitchburg, MA 01420-4142**

Syntax: **&CommonDir** (Programmers Macro)

Abstract: The common directory macro is a programmers macro, designed to pull the contact set path from the currently active GoldMine.

Example: **ddeRequest(nChannel, [&CommonDir])**

Returned: **SQLGoldMine:**

Syntax: **&Company**

Abstract: The **company** macro was designed to pull only the information contained in the Contact1.Company field. In most cases, one could simply have used Contact1.Company instead of the macro. This macro does trim off any trailing spaces in the field.

Prerequisite: Contact1.Company = Computerese Inc

Example: **&Company**

Returned: **Computerese Inc**

Syntax: **&Contact**

Abstract: The **contact** macro was designed to pull only the information contained in the Contact1.Contact field. In most cases, one could simply have used Contact1.Contact instead of the macro. This macro does trim off any trailing spaces in the field.

Prerequisite: Contact1.Contact = DJ Hunt

Example: **&Contact**

Returned: **DJ Hunt**

Syntax: **&Country**

Abstract: The **country** macro was designed to pull only the information contained in the Contact1.Country field. In most cases, one could simply have used Contact1.Country instead of the macro. This macro does trim off any trailing spaces in the field.

Prerequisite: Contact1.Country = USA

Example: **&Country**

Returned: **USA**

Syntax: **&DefaultRecordType**

Abstract: The **default record type** macro returns the default Record Type name, if one is employed. If none is employed, (default) is returned.

Example: **&DefaultRecordType**

Returned: **Customer**

Syntax: **&Dial1** (Programmers Macro)

Abstract: The **dial** phone**1** macro is a programmers macro designed to pull the Contact1. Phone1 field from the currently active GoldMine record. The returned phone number is formatted for dialing, based upon your current TAPI settings. If it is being used, any PreDial.ini settings are applied to the output.

Prerequisite: Contact1.Phone1 = (978)342-3333

Example: **ddeRequest(nChannel, [&Dial1])**

Returned: **T 1 978 3423333**

Syntax: **&Dial2** (Programmers Macro)

Abstract: The **dial** phone**2** macro is a programmers macro designed to pull the Contact1.Phone2 field from the currently active GoldMine record. The returned phone number is formatted for dialing, based upon your current TAPI settings. If it is being used, any PreDial.ini settings are applied to the output.

Prerequisite: Contact1.Phone2 = (978)342-3333

Example: **ddeRequest(nChannel, [&Dial2])**

Returned: **T 1 978 3423333**

Syntax: **&Dial3** (Programmers Macro)

Abstract: The **dial** phone**3** macro is a programmers macro designed to pull the Contact1.Phone3 field from the currently active GoldMine record. The returned phone number is formatted for dialing, based upon your current TAPI settings. If it is being used, any PreDial.ini settings are applied to the output.

Prerequisite: Contact1.Phone3 = (978)342-3333

Example: **ddeRequest(nChannel, [&Dial3])**

Returned: **T 1 978 3423333**

Syntax: **&DialFax** (Programmers Macro)

Abstract: The **dial fax** macro is a programmers macro designed to pull the Contact1.Fax field from the currently active GoldMine record. The returned phone number is formatted for dialing, based upon your current TAPI settings. If it is being used, any PreDial.ini settings are applied to the output.

Prerequisite: Contact1.Fax = (978)342-8867

Example: **ddeRequest(nChannel, [&DialFax])**

Returned: **T 1 978 3428867**

Syntax: **&EmailAddress**

Abstract: The **email address** macro was developed so that users could pull the Primary Contacts E-mail Address when doing e-mail merges, document merges and in other areas. This macro can also be employed to create a Filter to find those that do or do not contain E-mail Addresses.

Example:	**&EmailAddress**	**.not. empty(&EmailAddress)**
Returned:	**DJH@DJ-Hunt.com**	**.T.** if there is a Primary E-mail Address

Syntax: **&Fax**

Abstract: The fax macro is a macro designed to pull the Contact1.Fax field from the currently active GoldMine record. The fax number will be formatted. If the user has a need for the fax number as it is presented in GoldMine, then they must use the field directly instead of employing the macro.

Prerequisite: Contact1.Fax = (978)342-8867

Example:	**&Fax**	**Contact1->Fax**
Returned:	**1-978-342-8867**	**(978)342-8867**

Note

*Notice that the **&Filter** macro does not pull the Filter name, but, instead, actually pulls the syntax of the expression.*

Syntax: **&Filter** (Programmers Macro)

Abstract: The **filter** macro was designed to pull the currently active Filter from the current GoldMine session.

Example: **&Filter**

Returned: **trim(contact2->upw) > ' '**

Syntax: **&FirstName**

Abstract: The **first name** macro was developed to extract the first word from the Contact1. Contact field of the active record in the currently active GoldMine session.

Prerequisite: Contact1.Contact = Donald J. Hunt

Example: **&FirstName**

Returned: **Donald**

Syntax: **&FullAddress**

Abstract: The **full address** macro was designed to allow the user to place a formatted address field in merge documents, e-mail merge templates, or reports.

Prerequisite: Contact1.Address1 = 150 Pratt Road
Contact1.Address2 = Suite A
Contact1.Address3 =
Contact1.City = Fitchburg
Contact1.State = MA
Contact1.Zip = 01420
Contact1.Country = USA

Example: **&FullAddress**

Returned: **150 Pratt Road**
Suite A
Fitchburg, MA 01420
USA

Syntax: **&GetRoTabID** (Programmers Macro)

Abstract: The **get ro**w **tab id** macro was developed to find the currently active tab exposed on the currently active contact record in GoldMine. The tab numbers are 0 based and continue on up from there. The numbering sequence is based on the default legacy tab order.

Example: **&ddeRequest(nChannel, [&GetRoTabID])**

Returned: **5** indicating the Details tab

Syntax: **&GoldDir** (Programmers Macro)

Abstract: The **Gold**Mine **dir**ectory macro, is a programmers macro, designed to pull the path to the GoldMine legacy Base tables.

Example: **ddeRequest(nChannel, [&GoldDir])**

Returned: **SQLGoldMine:**

Syntax: **&LastFirstName**

Abstract: The **last first name** macro was developed to strip the last word and first word from the Contact1.Contact field, transpose the two, inserting a comma and space in between the two.

Prerequisite: Contact1.Contact = DJ Hunt

Example: **&LastFirstName**

Returned: **Hunt, DJ**

Syntax: **&LastName**

Abstract: The **last name** macro was designed to pull only the information contained in the Contact1.LastName field. In most cases, one could simply have used Contact1.LastName instead of the macro. This is not a macro that will strip the last word from the Contact1.Contact field. This macro does strip any trailing spaces from the value in the Contact1.LastName field.

Prerequisite: Contact1.LastName = Hunt

Example: **&LastName**

Returned: **Hunt**

Syntax: **&LicInfoLicTo** (Programmers Macro)

Abstract: Returns the Organization to whom this GoldMine is registered.

Example: **ddeRequest(nChannel, [&LicInfoLicTo])**

Returned: **Computerese**

Syntax: **&LicInfo_Contact** (Programmers Macro)

Abstract: Returns the name of the Contact to whom this GoldMine is registered.

Example: **ddeRequest(nChannel, [&LicInfo_Contact])**

Returned: **DJ Hunt**

Syntax: **&LicInfo_LicEmail** (Programmers Macro)

Abstract: Returns the email address as registered in GoldMine.

Example: **ddeRequest(nChannel, [&LicInfo_LicEmail])**

Returned: **DJ@DJHunt.US**

Syntax: **&LicInfo_Phone** (Programmers Macro)

Abstract: Returns the GoldMine registered phone number.

Example: **ddeRequest(nChannel, [&LicInfo_Phone])**

Returned: (978)342-3333

Syntax: **&LicInfo_Fax** (Programmers Macro)

Abstract: Returns the GoldMine registered fax number.

Example: **ddeRequest(nChannel, [&LicInfo_Fax])**

Returned: **(978)342-4567**

Syntax: **&LicInfo_Address1** (Programmers Macro)

Abstract: Returns the GoldMine registered information for the Address1 field.

Example: **ddeRequest(nChannel, [&LicInfo_Address1])**

Returned: **150 Pratt Road**

Syntax: **&LicInfo_Address2** (Programmers Macro)

Abstract: Returns the GoldMine registered information for the Address2 field.

Example: **ddeRequest(nChannel, [&LicInfo_Address2])**

Returned: **Suite A**

Syntax: **&LicInfo_City** (Programmers Macro)

Abstract: Returns the GoldMine registered information for the City field.

Example: **ddeRequest(nChannel, [&LicInfo_City])**

Returned: **Fitchburg**

Syntax: **&LicInfo_State** (Programmers Macro)

Abstract: Returns the GoldMine registered information for the State field.

Example: **ddeRequest(nChannel, [&LicInfo_State])**

Returned: **MA**

Syntax: **&LicInfo_Zip** (Programmers Macro)

Abstract: Returns the GoldMine registered information for the Zip field.

Example: **ddeRequest(nChannel, [&LicInfo_Zip])**

Returned: **01420-4142**

Syntax: **&LicInfo_Country** (Programmers Macro)

Abstract: Returns the GoldMine registered information for the Country field.

Example: **ddeRequest(nChannel, [&LicInfo_Country])**

Returned: **USA**

Syntax: **&LicUsers** (Programmers Macro)

Abstract: The **lic**ensed **users** macro is a programmers macro designed to pull the number of concurrent licensed users from the GoldMine Serial Number.

Example: **ddeRequest(nChannel, [&LicUsers])**

Returned: **25**

Syntax: **&LicUsersAvailable** (Programmers Macro)

Abstract: The **lic**ensed **users available** macro returns the number of users allowed to log in to the active GoldMine session.

Example: **ddeRequest(nChannel, [&LicUsersAvailable])**

Returned: **10**

Syntax: **&NameAddress**

Abstract: The **name** and **address** macro was designed to pull together, in label format, all of the information contained in the Contact, Company, Address1, Address2, Address3, City, State, Zip and Country fields of GoldMine. Any blank fields will be removed from the format automatically.

Prerequisite: Contact1.Company = Computerese
Contact1.Contact = DJ Hunt
Contact1.Address1 = 150 Pratt Road
Contact1.Address2 = Suite A
Contact1.Address3 =
Contact1.City = Fitchburg
Contact1.State = MA
Contact1.Zip = 01420
Contact1.Country = USA

Example: **&NameAddress**

Returned: **DJ Hunt**
Computerese
150 Pratt Road
Suite A
Fitchburg, MA 01420
USA

Syntax: **&NameTitleAddress**

Abstract: The **name**, **title**, and **address** macro was designed to pull together, in label format, all of the information contained in the Contact, Title, Company, Address1, Address2, Address3, City, State, Zip and Country fields of GoldMine. Any blank fields will be removed from the format automatically.

Prerequisite: Contact1.Company = Computerese
Contact1.Contact = DJ Hunt
Contact1.Title = GoldMine Guru
Contact1.Address1 = 150 Pratt Road
Contact1.Address2 = Suite A
Contact1.Address3 =
Contact1.City = Fitchburg
Contact1.State = MA
Contact1.Zip = 01420-4142
Contact1.Country = USA

Example: **&NameTitleAddress**

Returned:
DJ Hunt
GoldMine Guru
Computerese
150 Pratt Road
Suite A
Fitchburg, MA 01420-4142
USA

Syntax: **&NewRecID** (Programmers Macro)

Abstract: The **new re**cord **ID** macro was designed to return a new record id which could be used by programmers when they need to create a LOPRecID.

Example: **ddeRequest(nChannel, [&NewRecID])**

Returned: **5OL0TG0?0F_%(\%**

Syntax: **&Notes**

Abstract: The **notes** macro was designed to return a portion of the main contact notes, which are found under the Notes tab. Depending on where this macro is employed, the size of the Notes returned may vary.

Example: **&Notes**

Returned:
***** DJ *** 5/3/2010 12:55 pm**
Removing and reinstalling iTunes, QuickTime, and other software components for Windows Vista or Windows 7

***** DJ *** 10/4/2010 9:29 am**
Apple iPad Extended Care

Thru: August 8, 2012

Syntax: **&Phone**

Abstract: The **phone** macro was developed to pull the Contact1.Phone1 field information from GoldMine. This macro supports Additional Contacts when used in merge forms or e-mail templates. If the Additional Contacts, ContSupp.Phone field is blank, the macro will pull in the Contact1.Phone1 field from the Primary Contact.

Prerequisite:	Contact1.Phone1 = (978)342-3333	ContSupp.Phone = ""
Example:	**&Phone**	**&Phone**
Returned:	**(978)342-3333**	**(978)342-3333**

Tip

*You may not achieve the expected results from the **&Profile/&Profiles** macros. If you are having a problem achieving your desired results, you may want to remove any spaces from your Detail name. A common fix is to use the underscore in the Detail name.*

i.e. ***Beyond_Gold***

Syntax: **&Profile/&Profiles**

Abstract: The **profile** or **profiles** macro was developed to pull one Detail type from the active Contact record. It could pull a single Detail, or multiple Details of the same type. The term Profiles is a legacy hold over name for this information.

&Profile.DetailName.Reference.Flag
Retrieves the first Detail matching the DetailName and its Reference

&Profiles.DetailName.Reference.Flag
Retrieves all Details matching the DetailName and their References

The Flag can be either 2 or 4
2 Retrieves the DetailName and the Reference
4 Retrieves the DetailName, Reference, and all the extended fields

Example: **&Profile.Invoice..2**

Returned: **Invoice 1st Detail**

Example: **&Profiles.Invoice..4**

Returned: **Invoice 1st Detail Field1 Field2**
Invoice 2nd Detail Field1 Field2 Field3 Field4

Syntax: **&RecordType**

Abstract: The **record type** macro was developed to return the record type name or the current contact record as opposed to the &DefaultRecordTyp macro discussed earilier in this chapter.

Example: **&RecordType**

Returned: **Customer**

Syntax: **&SerialNo** (Programmers Macro)

Abstract: The **serial n**(**o**)umber macro was developed to extract the GoldMine serial number from the currently active GoldMine session.

Example: **ddeRequest(nChannel,4 [&SerialNo])**

Returned: **E-000690-519245-VAJQX-TJPC0-HS28M**

Syntax: **&SetRoTab#** (Programmers Macro)

Abstract: The **set ro**w **tab #**, where # = row tab id, macro was developed to allow the programmer to actively switch between tabs in the currently active GoldMine record.

Example: **ddeRequest(nChannel, [&SetRoTab5])**

Returned: **5** ID of currently set tab

Syntax: **&ShutDown** (Programmers Macro)

Abstract: The **shut down** macro was developed to close the currently active GoldMine session on the system on which this macro is run.

Example: **ddeRequest(nChannel, [&ShutDown])**

Returned: None **GoldMine closes**

Syntax: **&State**

Abstract: The **state** macro was designed to pull only the information contained in the Contact1.State field. In most cases, one could simply have used Contact1.State instead of the macro. This macro does trim off any trailing spaces at the end of the value in the State field.

Prerequisite: Contact1.State = MA

Example: **&State**

Returned: **MA**

Syntax: **&SysDir** (Programmers Macro)

Abstract: The **sys**tem **dir**ectory macro was designed to allow the programmer to pull the folder that contains the GoldMine exe's and dll's.

Example: **ddeRequest(nChannel, [&SysDir])**

Returned: **Y:\GoldMine**

Syntax: **&SysInfo** (Programmers Macro)

Abstract: The **sys**tem **info**rmation macro was designed to allow the programmer to pull all of the current GoldMine sessions system information.

Example: **ddeRequest(nChannel, [&SysInfo])**

Returned:
Product = GoldMine Premium Edition
License = E-000890-516764-PKT55-PJ686-1RQ6L
Versions = GoldMine 9.0.2.36, Windows NT 6.01.7601 Service Pack 1
GoldMine EXE = c:\program files (x86)\goldmine\gmw.exe
TEMP Directory = c:\users\admini~1\appdata\local\temp
System Files = \\192.168.1.125\apps\goldmine
GoldMine Files = SQLGoldMine: [ANSI]
Contact Files = SQLGoldMine: [ANSI]
Free Disk Space = C: 429,957 MB
Memory (RAM) = 8320 MB
Open Cursors = 53
Logged User = DJ [Master!] [SQL Login: sa]
CAL = [New RecID]
Contact1 = [New RecID]
Tx32 version = 801

Syntax: **&Title**

Abstract: The **title** macro was designed to pull only the information contained in the Contact1. Title field. In most cases, one could simply have used Contact1.Title instead of the macro, however, this macro trims off any trailing spaces in the value contained in the Title field.

Prerequisite: Contact1.Title = Owner

Example: **&Title**

Returned: **Owner**

Syntax: **&UserFullName**

Abstract: The **user full name** macro was designed to extract the users full name, as identified in the Users' Settings... User's Full Name field within GoldMine. A programmer might just pull this information out of the Users.Name field. This is a useful macro for document templates and e-mail templates.

Prerequisite: Users.Name = Donald J. Hunt

Example: **&UserFullName**

Returned: **Donald J. Hunt**

Syntax: **&UserName** (Programmers Macro)

Abstract: The **user name** macro was developed to pull the logged in userID from the currently active session of GoldMine.

Example: **ddeRequest(nChannel, [&UserName])**

Returned: **DJ**

Syntax: **&User_Var.<Variable Name>.<UserID>**

Abstract: The **user var**iable macro was developed to pull various user defined variables from the username.ini, however, over the years, it has been modified to extract information for other networked GoldMine users. Four of these variables are defined through the Preferences GUI. The users may add as many additional variables as they need.

Prerequisite:	Title = Owner	Email = DJ@DJHunt.US
Example:	**&User_Var.Title**	**&User_Var.Email**
Returned:	**Owner**	**DJ@DJHunt.US**
Prerequisite:	Title for Bubba = Bubba Guy	Email for Bubba = Bubba@DJ-Hunt.com
Example:	**&User_Var.Title.Bubba**	**&User_Var.Email.Bubba**
Returned:	**Bubba Guy**	**Bubba@DJHunt.US**

Syntax: **&Version** (Programmers Macro)

Abstract: The **version** macro was developed to pull the active GoldMine sessions version number.

Example: **ddeRequest(nChannel, [&Version])**

Returned: **9.0.2.36**

Syntax: **&WebSite**

Abstract: The **web site** macro was designed to extract the current contract records Primary WebSite from the ContSupp table.

Example: **&WebSite**

Returned: **http://www.DJHunt.US**

Syntax: **&Zip**

Abstract: The **zip** macro was designed to pull only the information contained in the Contact1.Zip field. In most cases, one could simply have used Contact1.Zip instead of the macro, however, this macro trims off excess trailing spaces from the value in the Zip field.

Prerequisite: Contact1.Zip = 01420-4142

Example: **&Zip**

Returned: **01420-4142**