# Converting Existing Programs to Layout Programs Using a Driver Program

A driver program collects data and loads it into a record structure. A layout program can then reference that record structure. There are several common uses of driver programs. For example, you may have a situation where the same query is used to retrieve data that is displayed in several different formats. While you could duplicate the query in a layout program for each output format, doing so would require you to modify each program if you needed to modify the query. To prevent this type of rework, you could place the query in a driver program and create a layout program for each format. Each layout program would use the driver program to retrieve the data. Then, if you needed to modify the query, you would only have to modify the driver program. Another common use of a driver program is to convert an existing program to use the Discern Layout Builder functionality. If the program is a simple one, it is most easily re-created as a layout program. The Insert From Clipboard option on the Add Query dialog box makes it easy to import an existing query, and the **Edit Code** button on the Add Queries dialog box can be used to re-create the code in the Reportwriter sections. However, if the existing program is large and complicated, it might be easier to convert it to a driver program that loads the data into a record structure, and then create a layout program that references that record structure. The following source code example creates a program that displays the number of orders placed each day for a date range that the user enters at run time.

The output of this simple program would look something like the following example:

```
Total Orders for:
     2007-05-30         400
     2007-05-31         279
     2007-06-01         111
     2007-06-03          21
     2007-06-04         323
     2007-06-05         298
     2007-06-06         272
```

If you want the data displayed in a graph instead of listed as numerical values, you can use the graphing functionality in Discern Layout Builder. Considering the simplicity of this example program, it would be very easy to create a layout program that selects the data, generates the counts, and displays the data as a graph. However, re-creating larger and more complex programs is not as easy. In those cases, it would be easier to convert the existing program into a driver program and then call the driver program from a layout program. To demonstrate the process, we will convert the example program above to a driver program and then call it from a layout program.

## Creating a Driver Program

A driver program takes the selected data and loads it into a record structure. Any tool can be used to create a driver program. Before we can go any further, the program needs to be created. Use the following steps to create the program in your environment.

1. Using DVDev, from the File menu, select **New**. The New dialog box opens.
2. From the File Type List, select Prompt Program.
3. In the Program Name box, enter **1_*your_initials*_Example_Driver** and click **OK**. The Discern Prompt Builder dialog box opens, with a single control for an output device.
4. Click **Add** to add the second prompt that asks the user for a start date.
5. On the **General** tab, set the following:

| | |
|---|---|
| **Prompt Display:** | Select the Beginning Date |
| **Prompt Name:** | Sdate |
| **Control Type:** | Date Time |

| | |
|---|---|
| **Prompt Type:** | String |

6. On the **Date/Time** tab, verify the Date Only option is selected and the Command Line Format is DD-MMM-YYYY.
7. In the Anchor date & time area on the **Calculate Default** tab, uncheck the Current date & time option. Enter a negative value in the Day or Month control, to make the default start date far enough in the past to allow the query to get orders that span several different days.

   The values you enter will depend on the data that exists in your environment. If you have a lot of data you can set the Day field to -7 to make the default start date one week ago. If you do not have a lot of orders, you could set the Month field to -3 to make the default start date three months ago.
8. Click **Add** to add a third prompt that asks the user for an end date.
9. On the **General** tab, set the following:

| | |
|---|---|
| **Prompt Display:** | Select the Ending Date |
| **Prompt Name:** | Edate |
| **Control Type:** | Date Time |
| **Prompt Type:** | String |

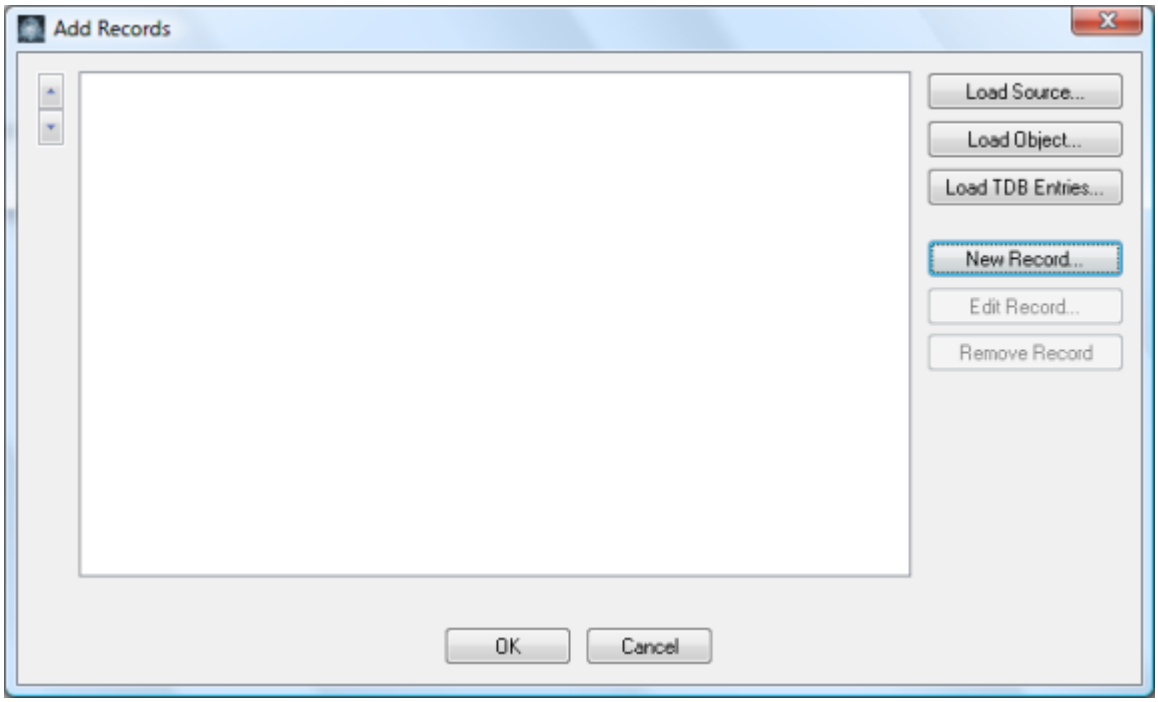
10. On the **Date/Time** tab, select the Date and Time option and set the Command Line Format to DD-MMM-YYYY HH:MM:SS.
11. Click **Save** to save the prompt form and close Discern Prompt Builder.
12. Copy the following select statement into your DVDev file. Place this code below the prompts and before the key words End Go.

13. Save and compile the source code file.
14. Execute the prompt program and verify it returns a count of the orders placed over several days.

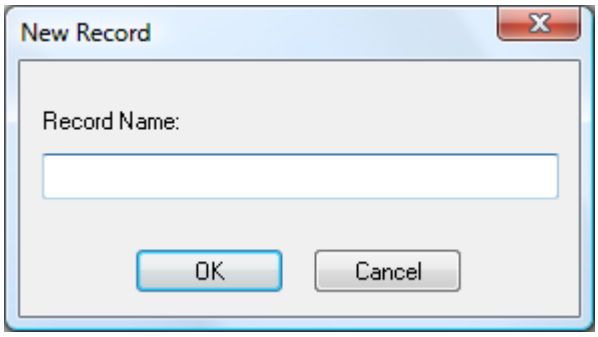
    You have now completed the steps necessary to create the prompt program to display a count of orders placed on a given day. We will assume this program was written in the past and we now want to use the functionality of the layout builder to add a graphical view of the data. To accomplish that, convert the program you just created to a driver program. Again, if the program does not already exist, it will be easier to create it as a layout program than to create a driver program and then call it from a layout program. The existing program displays the results of the select command using Reportwriter sections. We want to convert the existing program to a driver program that will load the results into a record structure instead of displaying them as output.

## Create the Record Structure

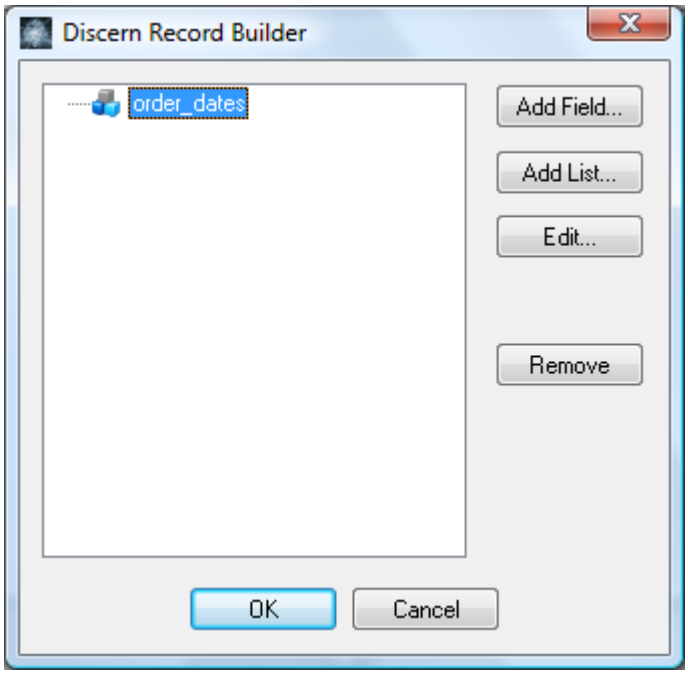
1. In your 1_*your_initials*_Example_Driver.prg file, place the cursor after the Prompt clause and before the Select command.
2. From the Tools menu, select **Record Builder**.
3. Select Yes at the prompt to add a new record. The Add Records dialog box opens.



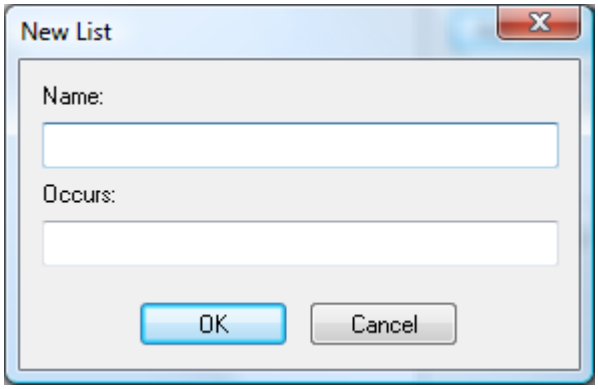

   The Add Records dialog box can be used to create new record structure definitions or to pull a record structure definition from an existing source code file, program object, or TDB entry into your source code file.
4. Click **New Record**. The New Record dialog box opens.



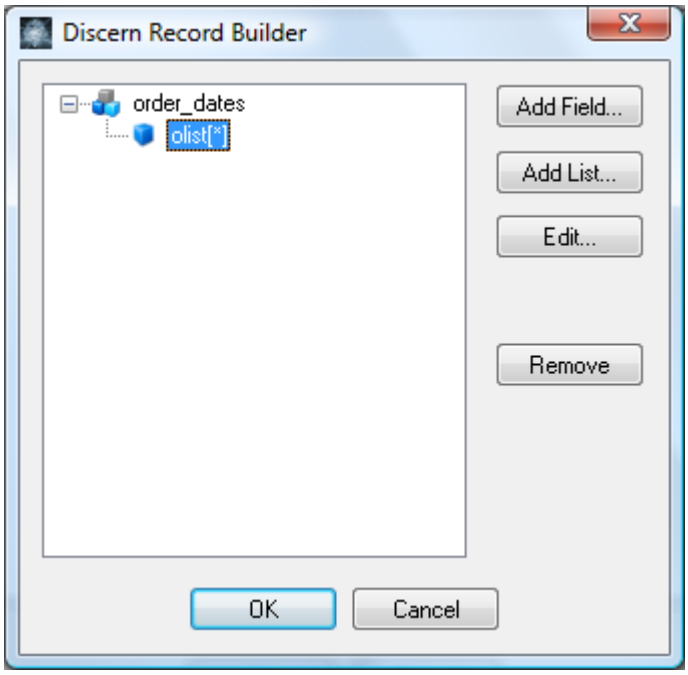

5. Enter **order_dates** as the Record Name and click **OK**. The Discern Record Builder dialog box is displayed.



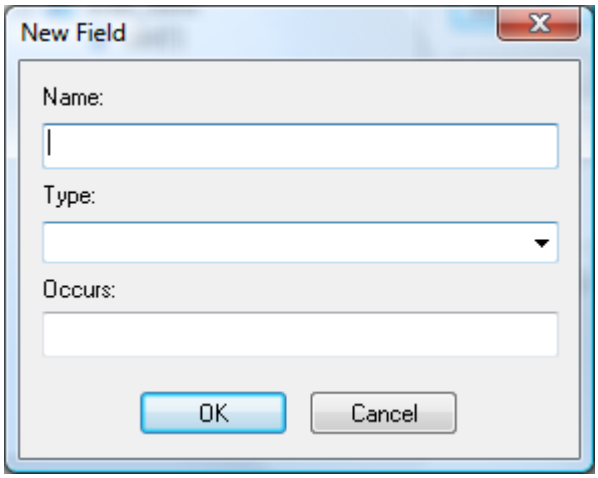

6. Click **Add List**. The New List dialog box opens.



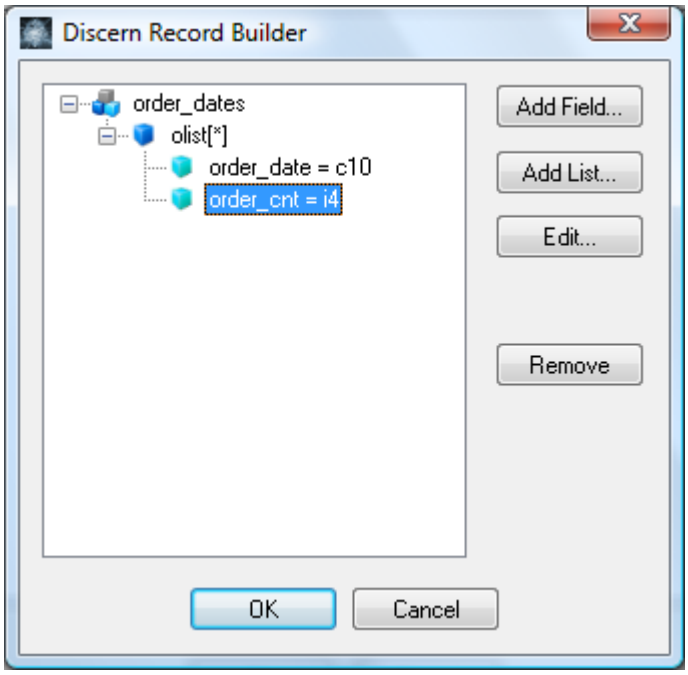

7. In the Name box, enter **olist** and in the Occurs box, enter *.
8. Click **OK** to close the Add List dialog box.
9. From the tree, select olist[*]. Your screen in Discern Record Builder should resemble the following:




10. Click **Add Field**. The New Field dialog box opens.




11. In the Name box, enter **order_date**.
12. In the Type box, enter **C10**.
13. Click **OK** to close the New Field dialog box.
14. Click **Add Field** to add a second field to the olist list.
15. In the Name box, enter **order_cnt**.
16. In the Type box, enter **i4**.
17. Click **OK** to close the New Field dialog box. Your Discern Record Builder dialog box is displayed similar to the following example:

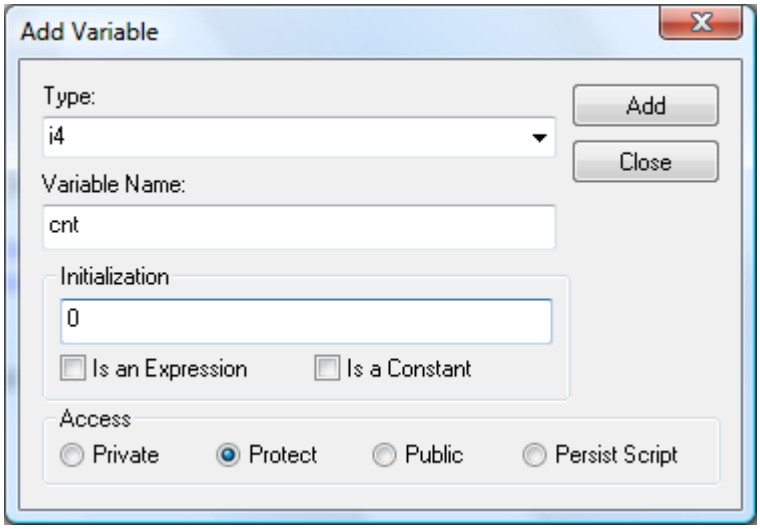



18. Click **OK** to close the Discern Record Builder dialog box.
19. Click **OK** to close the Add Records dialog box. The Record Builder added code similar to the following to your source code file:

## Create Needed Variables

In addition to the record structure you also need a counter variable to use for referencing the position of the olist in your order_dates record structure.

1. From the Tools menu, select **Add Variable**. The Add Variable dialog box opens.
2. From the Type list, select i4.
3. In the Variable Name box, enter **cnt**.
4. In the Initialization box, enter **0** (zero). Your Add Variable dialog is displayed similar to the following example:




5. Click **Add** to add the cnt variable.
6. Click **Close** to close the Add Variable dialog box. The following declare command is added to your source code:

**Note**
You have the option of entering Declare and Record commands directly in your source code file instead of using the Record Builder and Add Variable tools. However, using the tools eliminates the need to remember or look up the exact syntax for these commands.

## Store the Data in the Record Structure

We now need to modify the Reportwriter sections of the existing Select command to load the data into the record structure instead of displaying it.

1. Select the following code in your 1_*your_initials*_Example_Driver.prg file:

2. From the Edit menu, select **Extras** > **Comment Block** to comment out the selected code.
3. Place the following code directly below the code you commented out in the preceding step.

   Your complete 1_*your_initials*_Example_Driver.prg file now contains source code that is very similar to the following example:

   Since the driver program is loading data into a record structure instead of returning data for display, the prompt for an output device and selecting into that device is no longer needed. Therefore the output device prompt could be deleted and the Select Into $outdev could be modified to Select Into "NL:". Another option is to leave the prompt for the output device and the Select Into $outdev as is, and pass "NL:" as the first parameter when the driver program is called. For now we will use the second option. A start date in the format DD-MMM-YYYY and an end date in the format DD-MMM-YYY HH:MM:SS also need to be passed as parameters when the driver program is called.

4. Save and compile the source code file.

# Calling a Driver Program from a Layout Program

Once a driver program that loads data into a record structure has been created, it can be called from a layout program using **Set Layout Driver** from the Tools menu in the Layout Builder. In addition to setting the layout driver, the layout program needs to: Prompt the user for any input that is needed at run-time, define the record structure, access the data that is in the record, and display the data in the appropriate format.

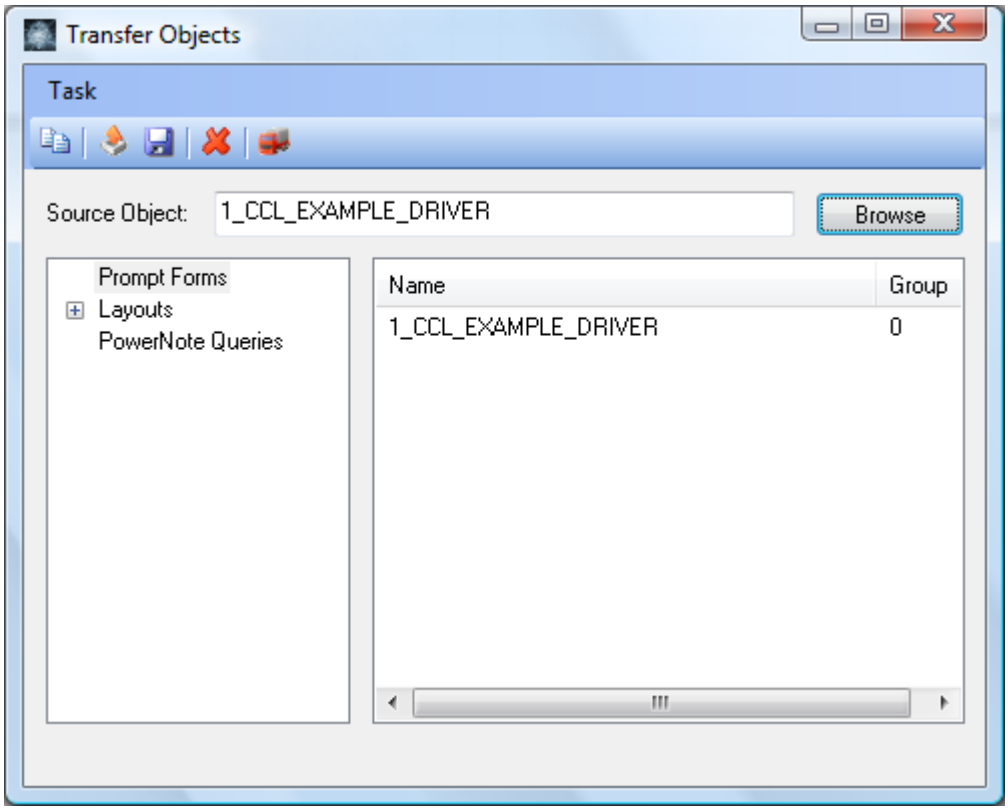
## Creating a New Layout Program To Use the Driver Program

1. Using DVDev, from the File menu, select **New**.
2. From the File Type list, select **Layout Program**.
3. In the Program Name box, enter **1_*your_initials*_Layout_Driver** and click **OK**. The New Layout Program dialog box opens.
4. Verify the Standard Layout option for the Report Layout and the PostScript option for the Output Type are selected and click **Next**.
5. Keep the defaulted values for the Paper Size and click **Finish**. A layout with a single section named DetailSection is created.

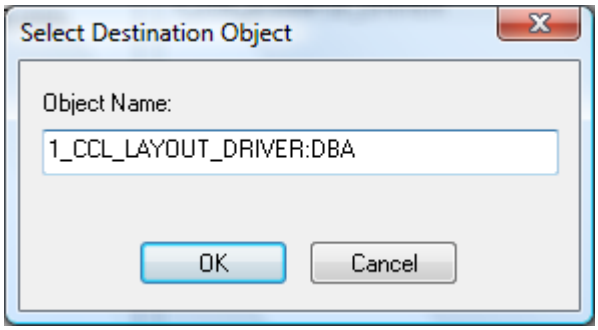
## Prompt for User Input

The layout program needs to prompt the user for any input that is required at run time. Since the driver program we are planning to use is expecting a start date and an end date, our layout program needs to prompt for these dates. The dates the user selects at the prompts are passed to the driver program when it is called. A prompt form can be created for the layout program just like it can be created for any other program using the prompt builder. However, because the program we converted to a driver program already has a prompt form, we can copy that prompt form for use by the layout program.

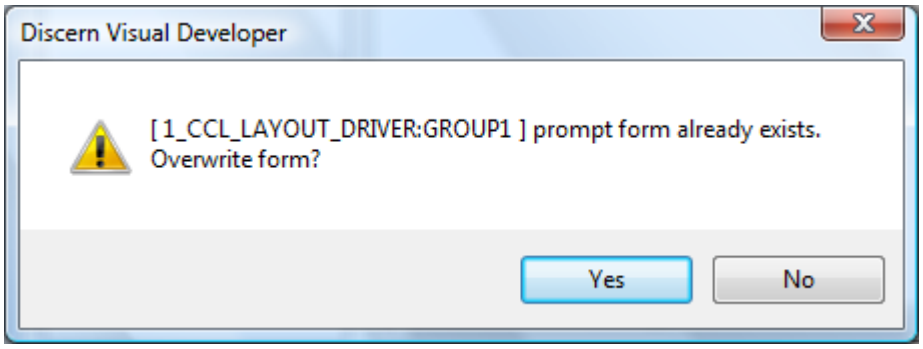
1. From the Tools menu, select **Transfer Objects**. The Transfer Objects dialog box opens.
2. In the Source Object box, enter **1_*your_initials*_Example_Driver** and click **Browse**. Your prompt form opens in the search result box.




3. Right-click your prompt form and select **Copy Object**. The Select Destination Object dialog box opens.

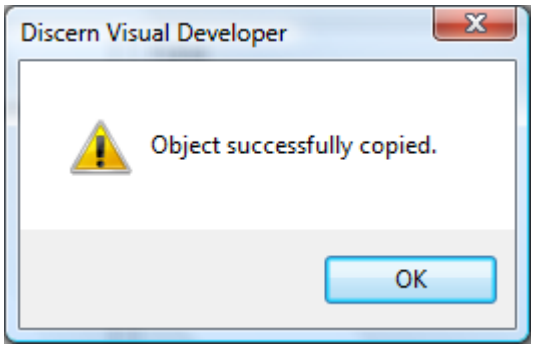



4. In the Object Name box, enter **1_*your_initials*_Layout_Driver** and click **OK**. A warning message similar to the following example is displayed:




When you created the layout program earlier in this exercise, a default prompt form with an output device control was created for your program.

5. Click **Yes** to overwrite the default form with the new copy. A message similar to the following example is displayed:

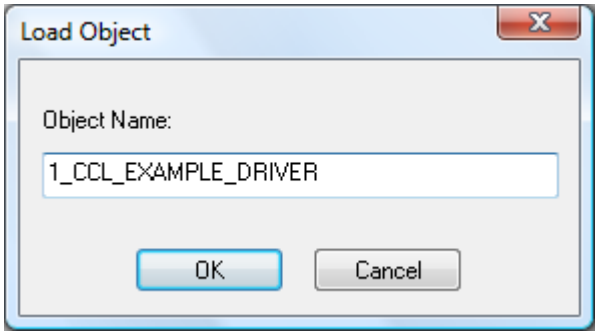



6. Click **OK** and close the Transfer Objects dialog box.

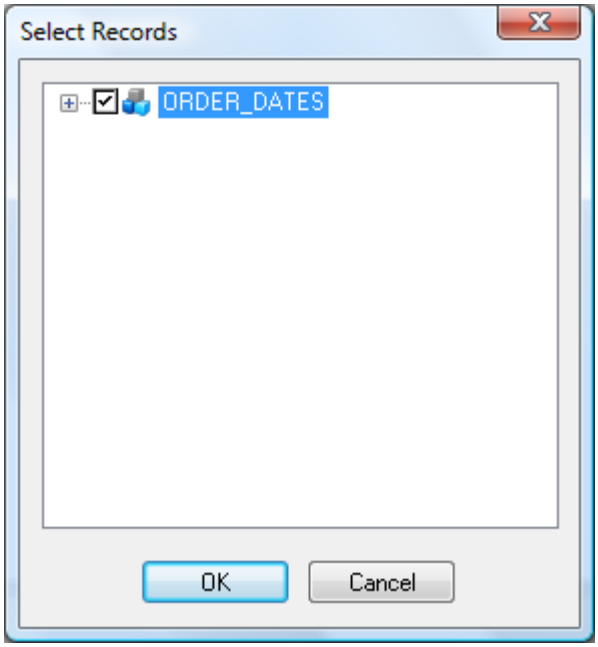
## Define the Record Structure

The layout program executes the driver program to load data into a record structure. For the layout program to format the data from the record structure, the record structure needs to be known in the layout program and in the driver program. Since the layout program will call the driver program, the easiest way to have the record structure known in both the layout and driver programs is to have the layout program create the record structure.

1. From the Tools menu, select **Record Builder**. The Add Records dialog box opens. You used the Add Records dialog box previously to create the record structure in the driver program. Since the record structure definition already exists in the driver program, you can use Load Object to have the Record Builder pull the definition from your 1*_your_initialsExample_Driver program into your 1your_initials*_Layout_Driver program.
2. Click **Load Object**. The Load Object dialog box opens.




3. In the Object Name box, enter **1*_your_initialsExample_Driver*** and click **OK**. The Select Records dialog box opens. Each record structure defined in the program object is displayed in the Select Records dialog box. Since your 1*your_initials*_Example_Driver program only has one record structure definition, your Select Records dialog box will resemble the following example:




4. Select Order_Dates and click **OK**. Your selection is displayed in the Add Records dialog box.
5. Click **OK** to close the Add Records dialog box.

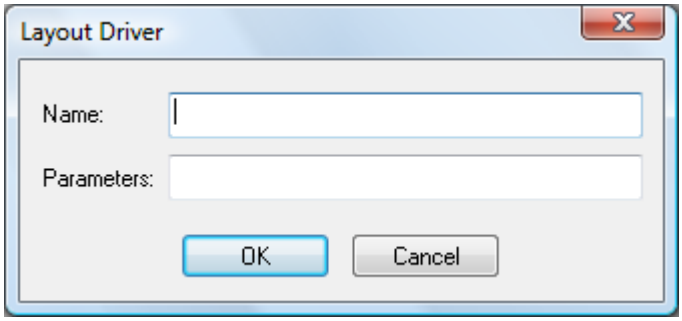
   Using the Add Records dialog box to load the Order_Dates record from your 1*_your_initials*_Example_Driver program creates a record command in the

layout code. At this point you have the record command to create the Order_Dates record in both the 1*_your_initials*Example_Driver and the 1 *your_initials*_Example_Driver programs. Since the layout program calls the driver program, it is not necessary to define the record structure in the driver program; by default it will be known in the driver. Having the record structure definition in both the driver program and the layout program will result in a warning that the record already exists when the driver program is executed. To prevent this warning you can delete the record definition from the driver program.

## Calling the Driver Program

Once a driver program has been created, it can be called or executed from a layout program by selecting **Set Layout Driver** from the Tools menu. Setting a layout driver will cause the layout program to execute the driver program.

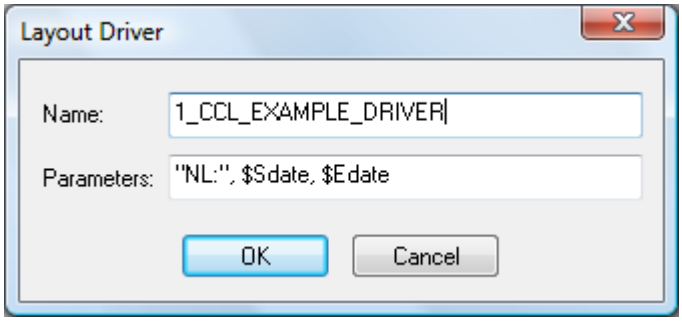
1. From the Tools menu, select **Set Layout Driver**. The Layout Driver dialog box opens.




The Layout Driver dialog box enables you to enter a program name in the Name box and any values that need to be passed to the program as parameters in the Parameters box. The example driver program you created previously used three prompts; one for output device, one for a starting date, and one for an ending date. When your layout program executes the driver program, it needs to pass parameters that will be assigned to each of the prompts in the driver program. Earlier, you created prompts for the layout program by copying the prompts from the driver program. The values that the user enters at the prompt for the start date and end date can be passed to the driver program. Since you modified the driver program to load data into a record structure, it needs to select into the null device ("NL:") instead of the output device that the user enters at the prompt. This can be accomplished two different ways. First, you could remove the prompt for the output device from the driver program, and modify the select into clause from Select INTO $OutDev to Select INTO "NL:". The second option is to simply pass "NL:" as the first parameter when you set the layout driver.

2. In the Name box, enter **1*_your_initials*_Example_Driver**.

   If your 1*_your_initials*_Example_Driver program, is a cclgroup1 object, you will need to append :group1 to the program name when you enter it in the Name: box. Use CCLPROT to determine if your program is a cclgroup1 or cclgroup0 (DBA) object.
3. In the Parameters box, enter **"NL:", $Sdate, $Edate**. Your Layout Driver dialog box should look similar to the following example:

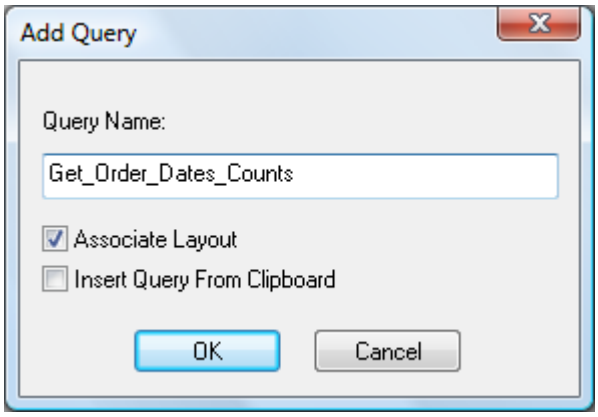



4. Click **OK** to close the Layout Driver dialog box.

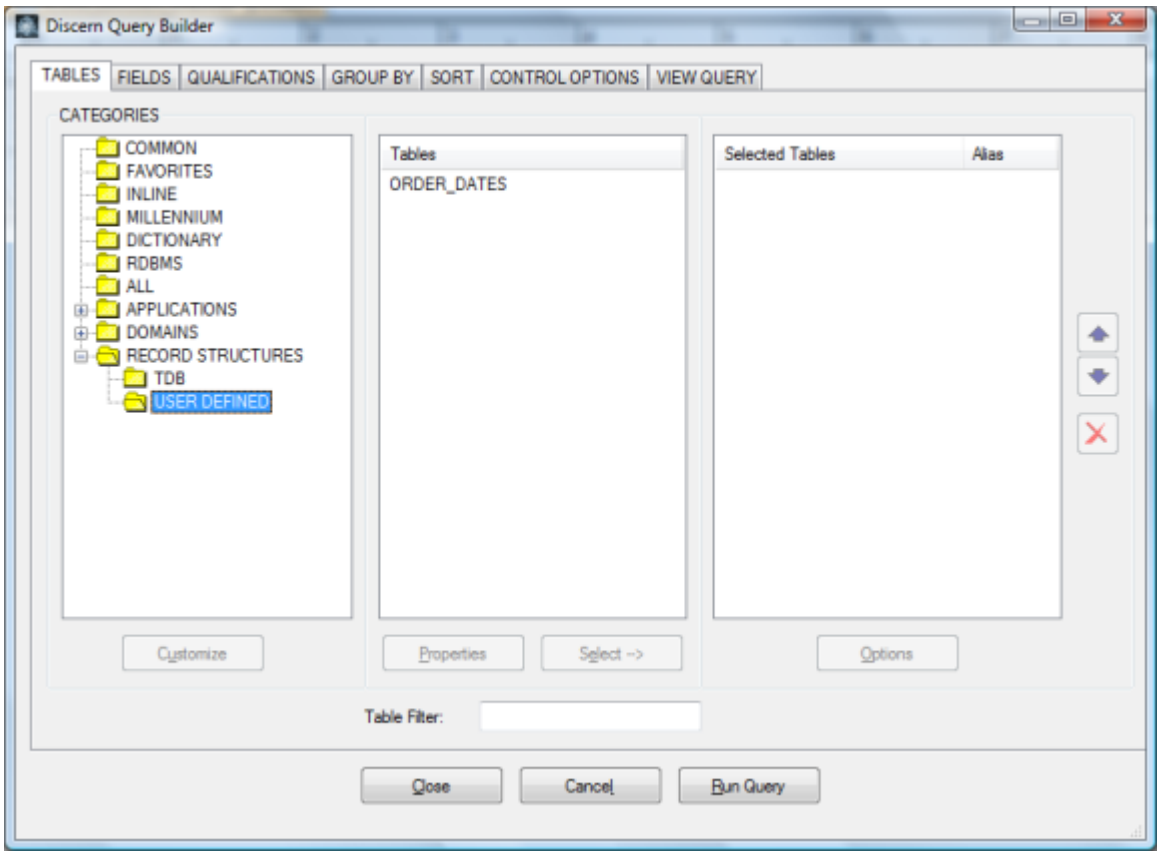
## Accessing and Displaying the Data From the Record Structure

Since the layout program defines the record structure and calls the driver program to load it, the record structure is known to the layout program and items in the record structure can be used as the source for items that are placed on the layout. The driver program is loading both a list of dates and a count of the number of orders placed on that date into the Order_Dates record structure. We want our layout program to display the date and count in text format and also create a graph that displays the number of orders per day. We therefore need to access each of the values in the record structure list. A simple way to access all the values in the record structure list is to use a query to assign the values in the list to select expressions.

1. From the Tools menu, select **Query Builder**. The Add Queries dialog box opens.
2. Click **Add**. The Add Query dialog box opens.
3. In the Query Name box, enter **Get_Order_Dates_Counts** and select the Associate Layout option. Your Add Query dialog box should look similar to the following example:

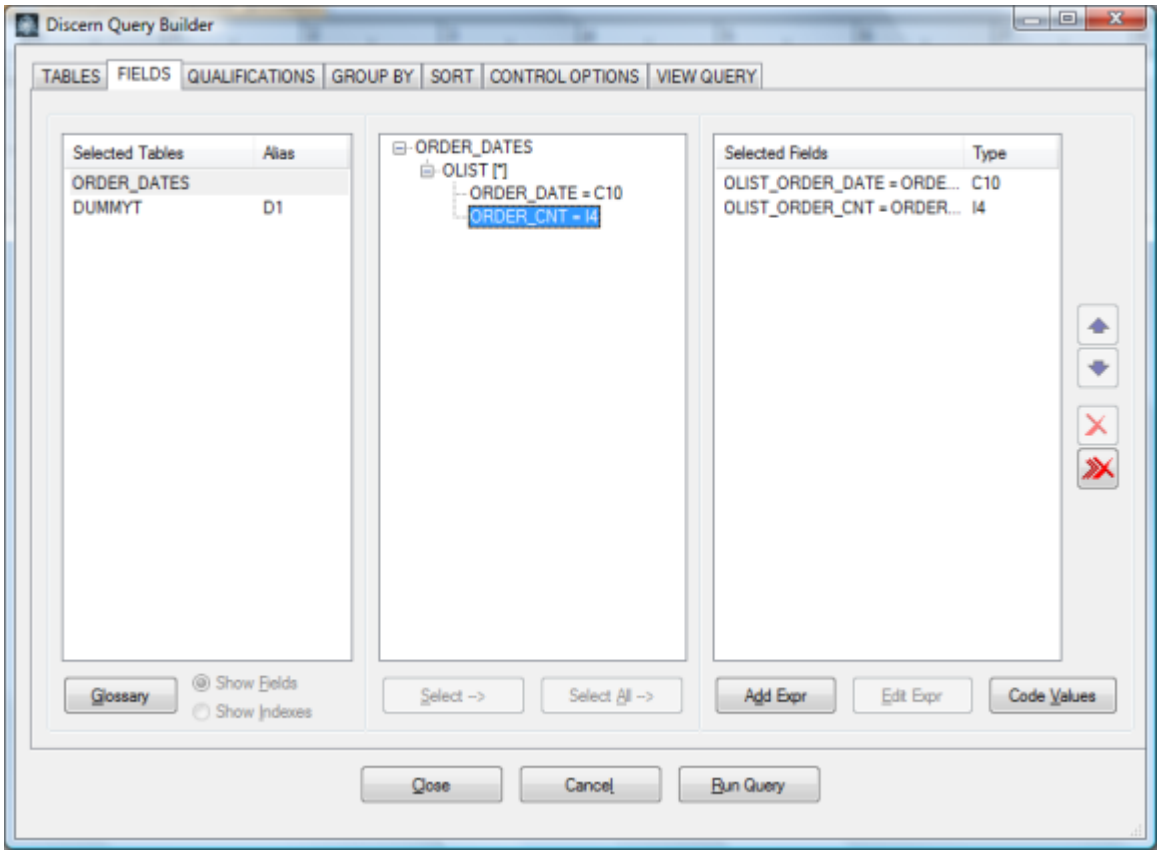



4. Click **OK**. The Discern Query Builder window opens.
5. In the **Tables** tab, expand the Record Structures tree and select the User Defined category. The ORDER_DATES record structure is displayed in the Tables column. Your Discern Query Builder window should look similar to the following example:

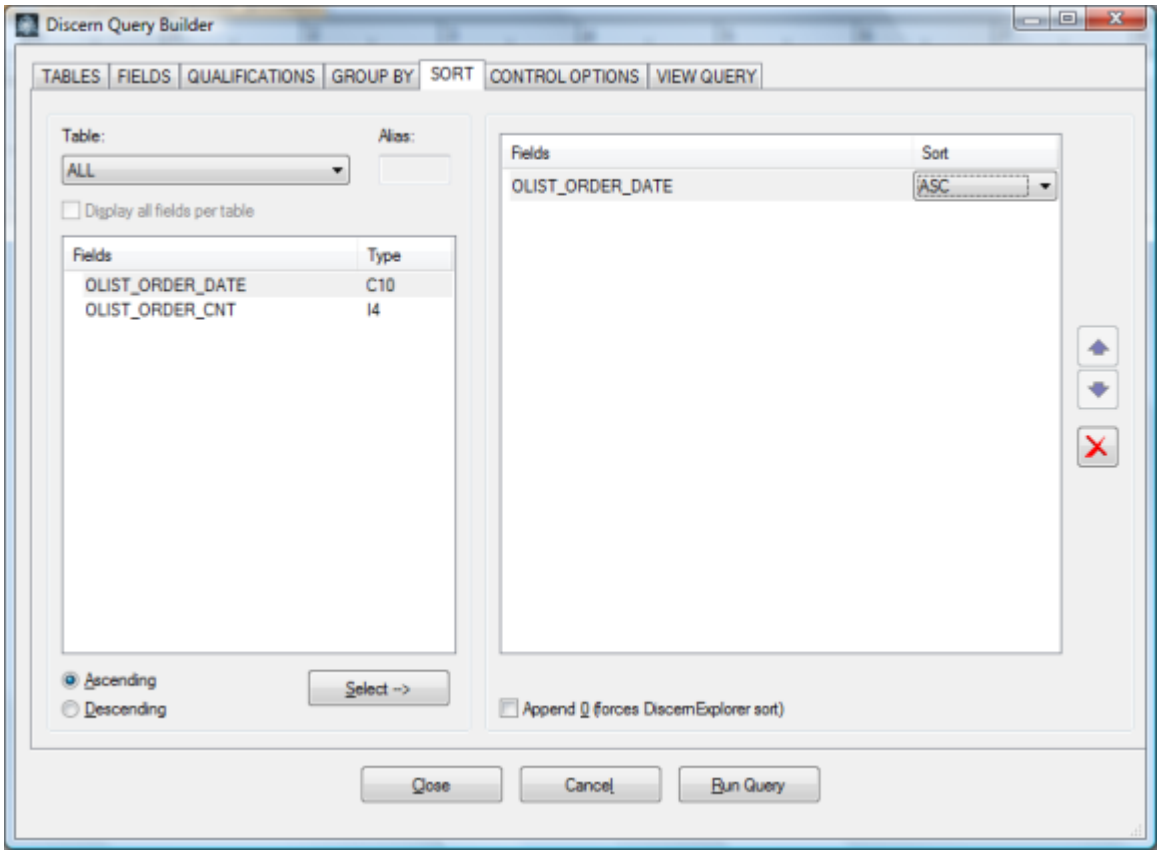



6. Double-click the ORDER_DATES record structure to move it to the Selected Tables column.
7. Click the **Fields** tab, expand the Olist tree, and double-click the ORDER_DATE and ORDER_CNT items to add them to the Selected Fields list. Double-clicking a record structure list item adds the DUMMYT table to the **Tables** tab and creates expressions in the Selected Fields column using *dummyt_alias*.seq to reference each position of the record structure list. Creating an expression for each item in a record structure makes it easier to

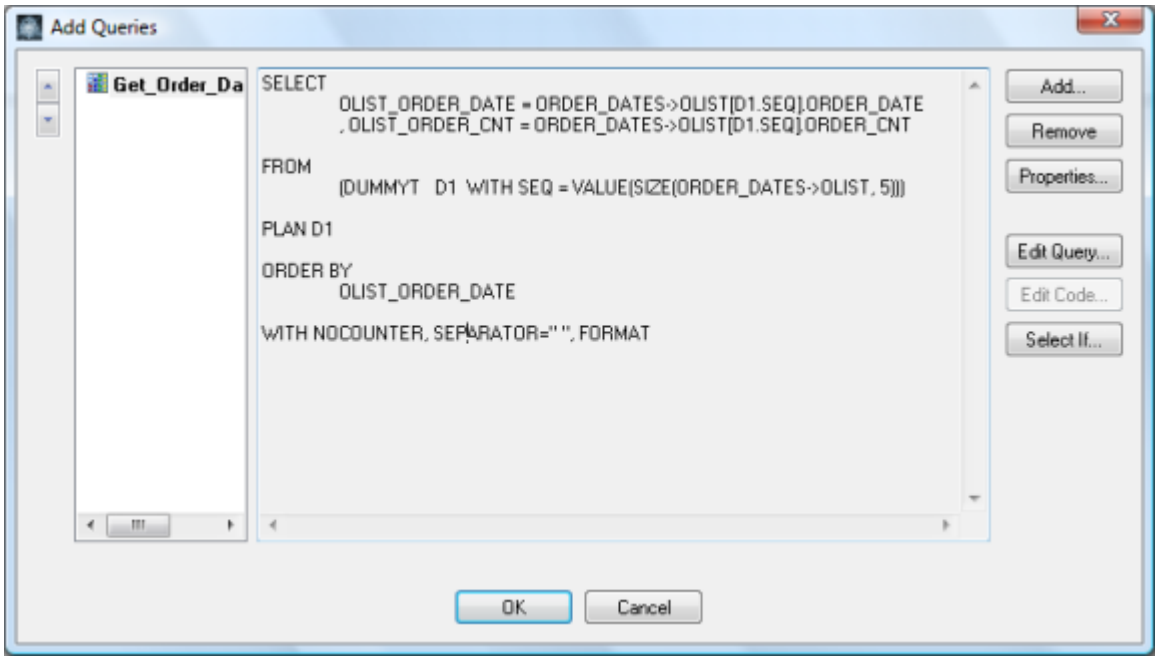
reference the items in your layout. Your Discern Query Builder dialog box should look similar to the following example:



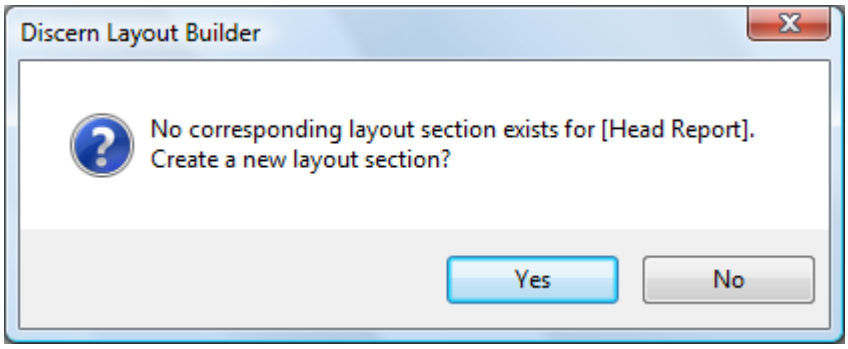

8. Click the **Sort** tab and sort by the OLIST_ORDER_DATE expression. Your Discern Query Builder dialog box should look similar to the following example:



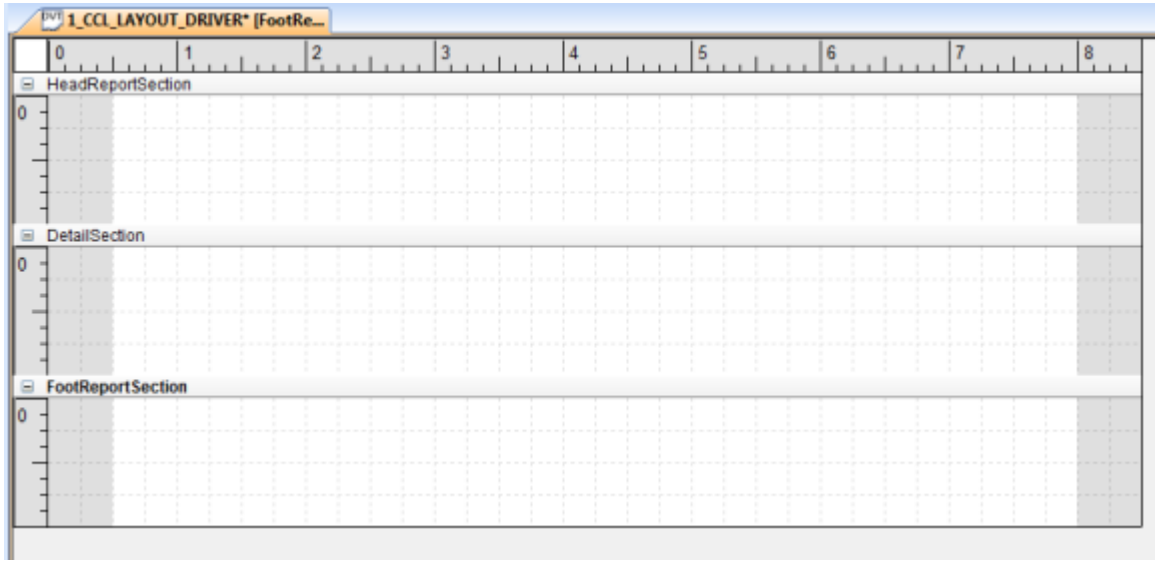

9. Click **Close**. The Add Queries dialog box opens.




10. Click **OK** to close the Add Queries dialog box. The Layout Workspace dialog box opens.
11. Check the Head Report Reportwriter Sections check box. The No corresponding layout section exists for [Head report]. Create a new layout section? message shown below opens.




12. Click **Yes** to add a HeadReportSection Layout Section.
13. Check the Detail and Foot Report Reportwriter Sections check boxes.
14. Click **Yes** at the message to create a new FootReportSection Layout Section. Your layout should resemble the following example:

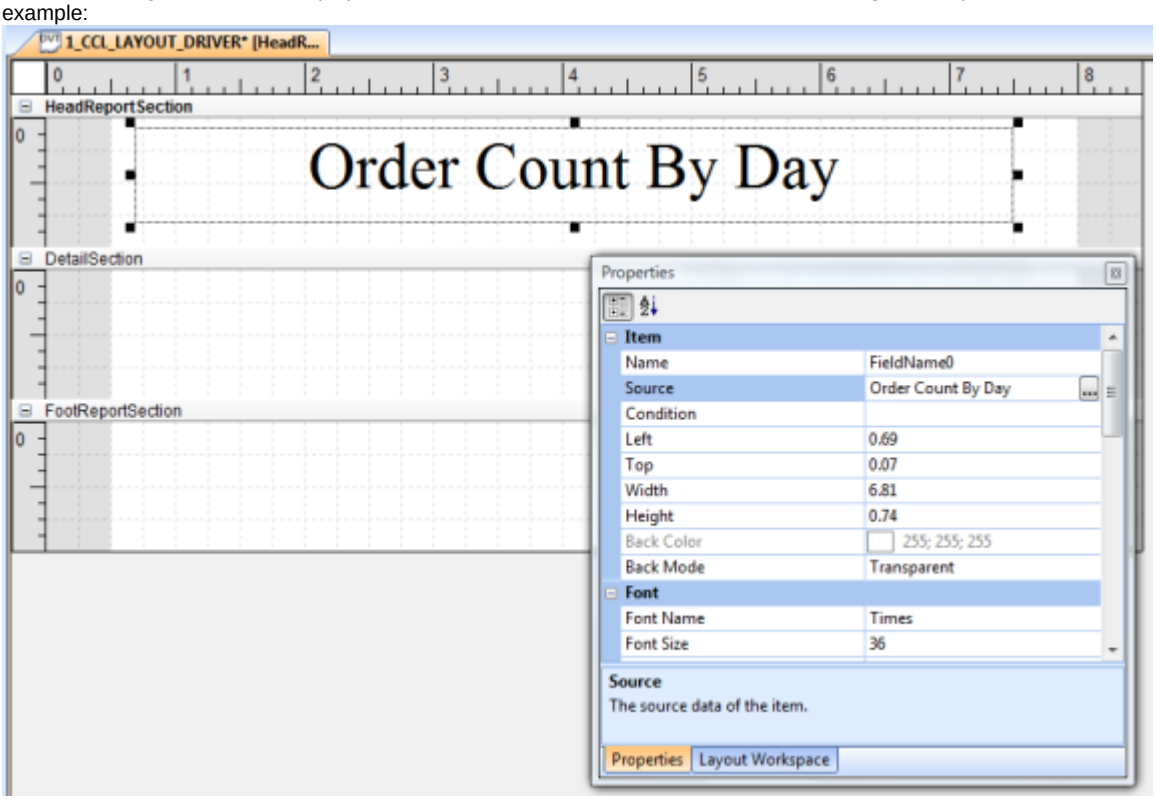



If your layout is displayed differently, you might want to modify the display using the View menu. For the example above, the following items are selected on the View menu: Properties, Horizontal Ruler, Vertical Ruler, Grid, Margins, and Section Title Bar.

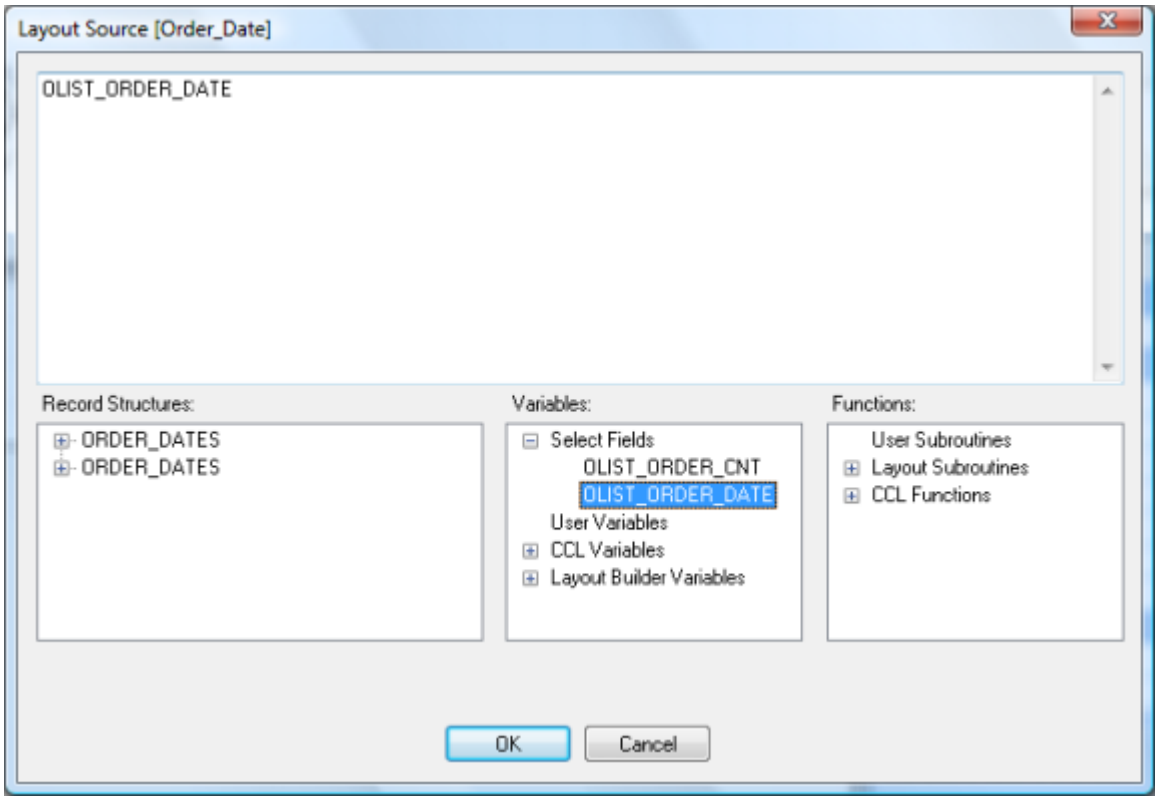
15. Use the Label Tool  to add a report title Order Count By Day to the HeadReportSection.
16. Use the Formatting toolbar to set the following values:

1. The font to Times
2. The font size to 36
3. Center the text

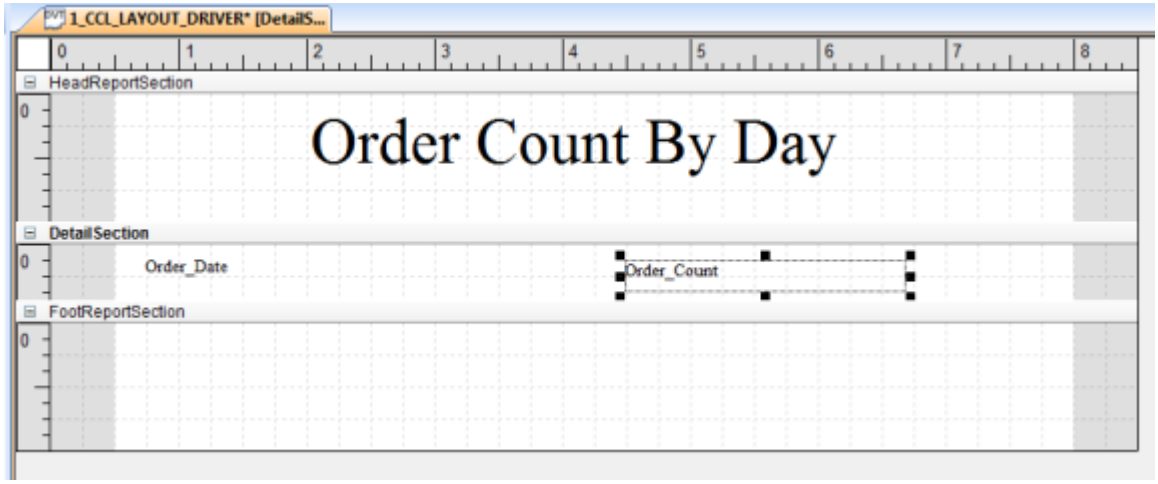

If the Formatting toolbar is not displayed, from the View menu, select **Toolbars** > **Formatting**. Your layout should resemble the following example:




17. Use the Text Tool  to place a text item in the DetailSection.
18. In the Properties dialog box, modify the Name property to **Order_Date**.
19. Click in the Source box to activate the ellipsis  button. Use the ellipsis  button to open the Layout Source [Order_Date] dialog box.
20. Expand the Select Fields tree in the Variables column and double-click the OLIST_ORDER_DATE expression to add it as the Layout Source. Your Layout Source [Order_Date] dialog box should look similar to the following example:

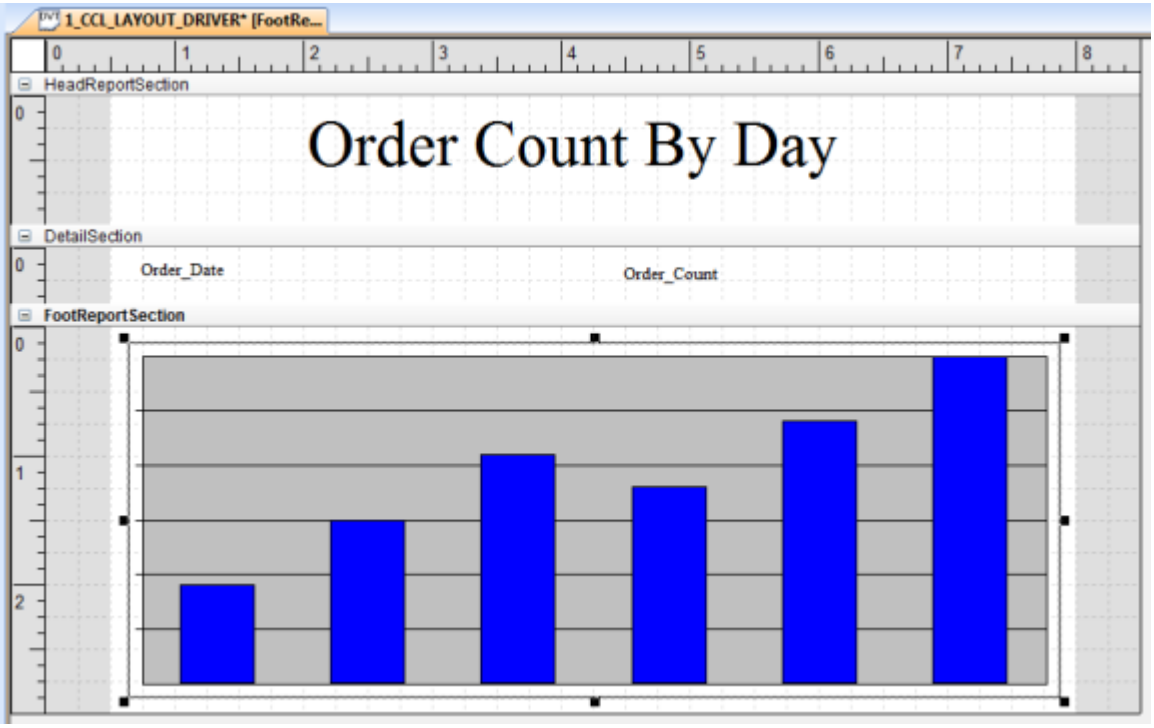



21. Click **OK** to close the Layout Source [Order_Date] dialog box.
22. Use the Text Tool  to place a text item to the right of the Order_Date in the DetailSection.
23. In the Properties dialog box, modify the Name property to **Order_Count**.
24. Click in the Source box to activate the ellipsis.  button. Use the ellipsis  button to open the Layout Source [Order_Date] dialog box.
25. Expand the Select Fields tree in the Variables column and double-click the OLIST_ORDER_CNT expression to add it as the Layout Source.
26. Click **OK** to close the Layout Source [Order_Date] dialog box.
27. Slowly move your pointer over the FootReportSection title bar. At the top of the title bar, the pointer changes to the vertical resize pointer. Use the vertical resize to move the bottom of the DetailSection up to just underneath the items you just added. Your layout should look similar to the following example:

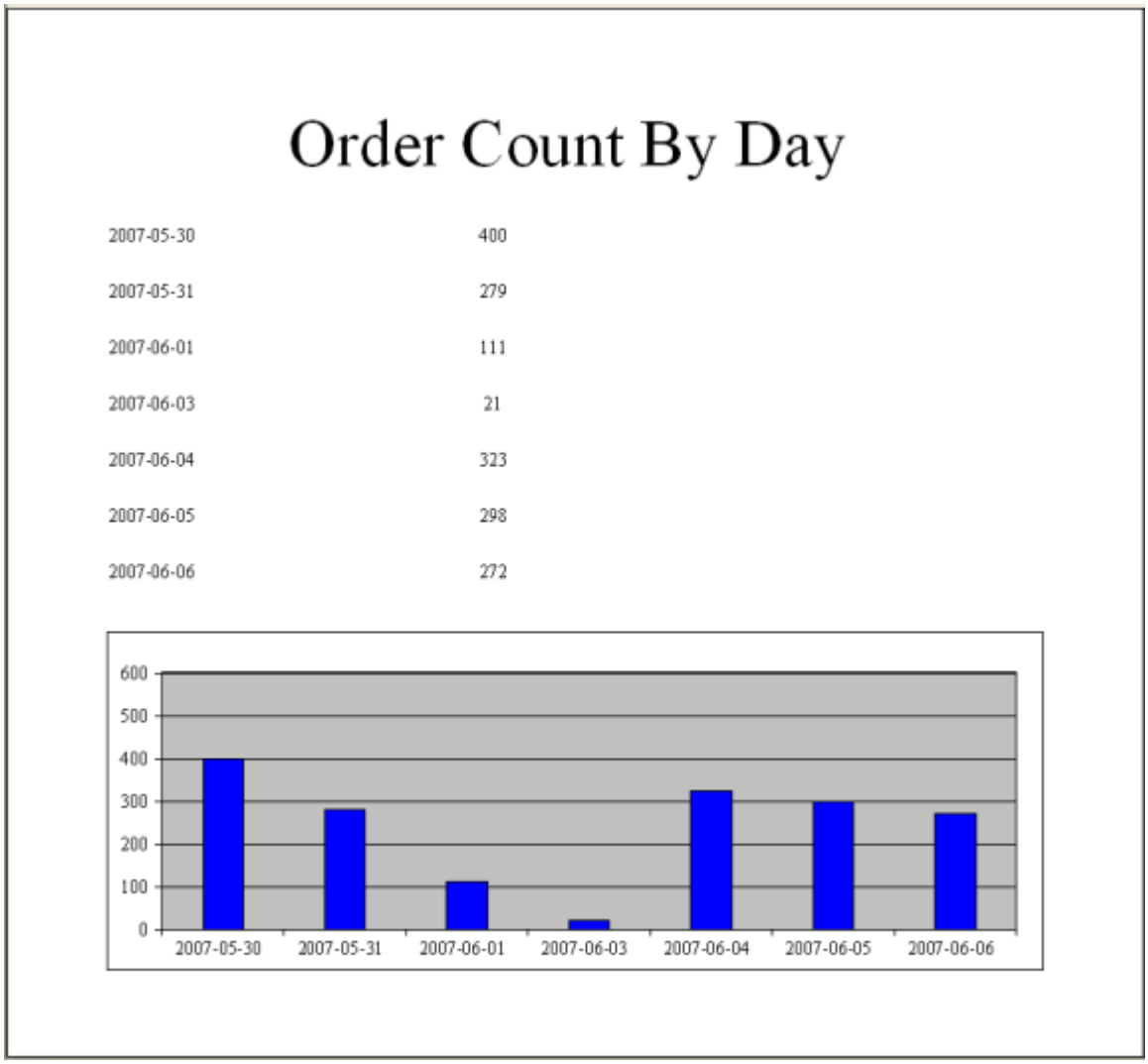



28. Use the vertical resize to move the bottom of the FootReportSection down to make the total height of the section three inches.
29. Use the Graph Tool  to create a graph item that covers most of the FootReportSection.
30. In the **Data Source** tab of the Graph Properties dialog box, perform the following actions:
    1. Select Get_Order_Dates_Counts as the query.
    2. Use OLIST_ORDER_DATE as the Axis Field and Axis Label.
31. In the **Series** tab of the Graph Properties dialog box, perform the following actions:
    1. Modify the Name property of Series 1 to **Order Count**.
    2. Enter **OLIST_ORDER_CNT** in the Data Values field, or click the ellipsis  button and select OLIST_ORDER_CNT from the Select Fields list.

32. Click **OK** to close the Graph Properties dialog box. Your layout should look similar to the following example:




33. Save the layout.
34. Execute the layout by selecting Run "1_*your_initials*_Layout_Using_Driver" from the Build menu or by clicking the **Run Prompt Program** button on the toolbar.
35. Use MINE as the output device.
36. Select starting and ending dates that allow you to qualify orders placed over several different days. Your output should look similar to the following example:




## Displaying Data From Multi-Level Record Structures

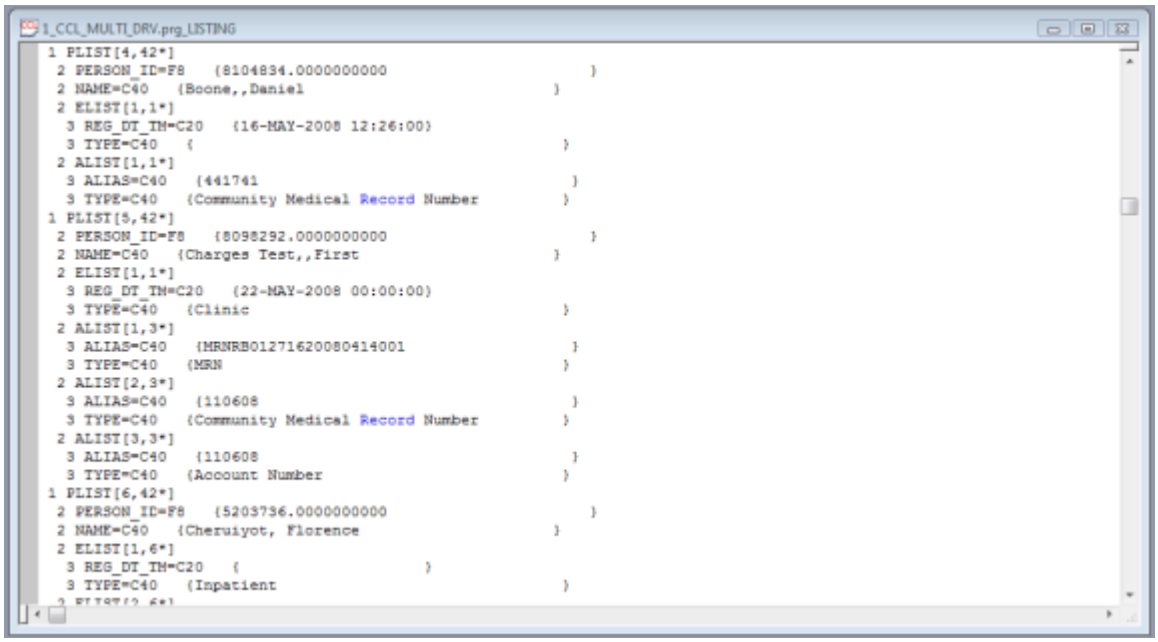
Multi-level record structures are often used to temporarily store data gathered from several select statements. Once the data has been gathered, it is formatted for display. The following example demonstrates how to use a driver program to load data into a multi-level record structure and how to use a layout program to format the data for display.

1. Using DVDev, from the File menu select **New** and create a new program named 1*_your_intials*_multi_drv.prg.

   The following code creates a multi-level record structure named Person_Enc_Alias, which stores information about people, their encounters, and their aliases. Two selects are used to get this information and load it into the record structure.

2. Copy and paste the above code above into the 1*_your_intials*_multi_drv.prg file you created in Step 1.
3. Enter **1*_your_intialsmulti_drv GO*** _at the end of your 1*your_intials*_multi_drv.prg file. Ensure this command is placed after the End Go.
4. Save and Include/Compile your 1*_your_intialsmulti_drv.prg file to create and execute your 1your_intials*_multi_drv program.
5. Press CTRL+L, click **Listing**, or from the View menu select **Listing** to open the listing file.

   The listing file is created automatically by DVDev when you Include/Compile a file. The Call Echorecord(person_enc_alias) command at the end of your 1*_your_intials*_multi_drv program echoes the contents of the person_enc_alias record structure to the listing file. As you scroll through the listing file, you will see first the commands that were compiled and then the output of the Call Echorecord command.
6. Review the output of the Call Echorecord command in the listing file and verify your 1*_your_intials*_multi_drv program is creating the record structure and populating it with the names and person IDs of people, the encounter type and registration date and time of their encounters, and their aliases and alias types. Your listing file should look similar to the following example:



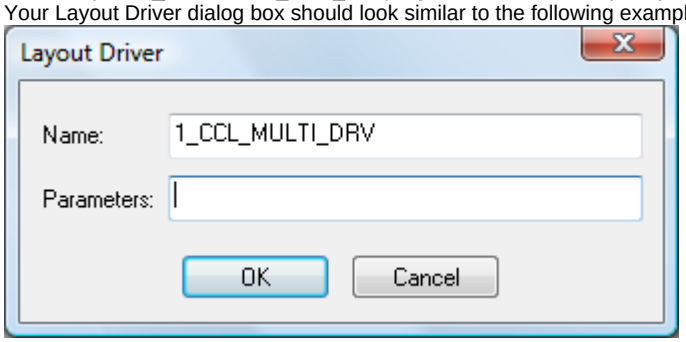

If your Person_Enc_Alias record structure is empty, you might need to modify the qualifications on the first select command to allow it to find some people with encounters and aliases.
Each person stored in the record structure can have multiple encounters and multiple aliases. We want to use Layout Builder to display this information in a user-friendly format. To do this, we will need to traverse the PLIST and display the person information. On each person we will need to traverse first the ELIST to display the encounter information and then the ALIST to display the alias information. Joins to the DUMMYT table can be used to traverse the level 1 PLIST and one of the level 2 lists (ELIST or ALIST). A For loop can then be used to traverse the other Level 2 list. The following source code example shows how these methods could be used to display the data from the Person_Enc_Alias record structure in a simple ASCII report.

---

We will now begin the process of creating the same code to traverse the record structure lists; however, we will use the Layout Builder functionality to display the data instead of the simple row and col commands shown in the ASCII version above. Using the Layout Builder in this way allows you to create code using all of the Layout Builder functionality that would otherwise be difficult to create if we were writing the program in a text editor.

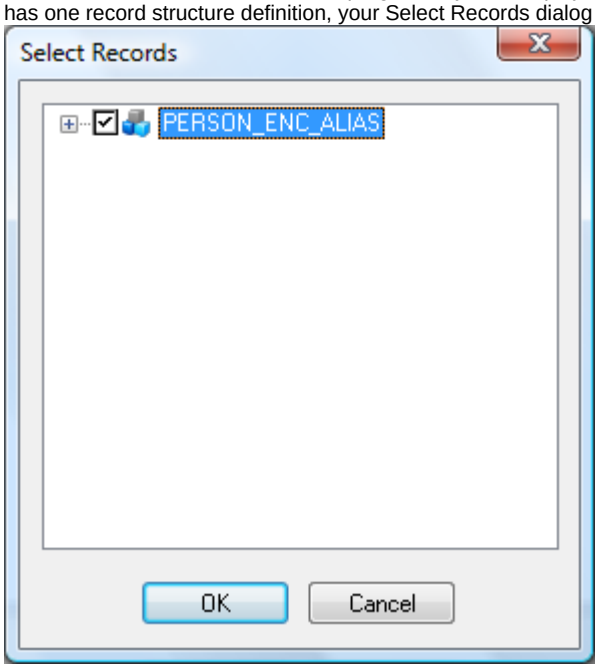
7. Using DVDev, from the File menu, select **New**.
8. From the File Type list, select **Layout Program**.
9. In the Program Name box, enter **1*_your_initials*_Multi_Layout** and click **OK**. The New Layout Program dialog box opens.
10. Verify the Standard Layout option for the Report Layout and PostScript option for the Output Type are selected and click Next. The Paper Size dialog box is displayed.
11. Keep the defaulted values for the Paper Size and click **Finish**. The previous two dialog boxes, New Layout Program and Paper Size, populate the basic properties of the layout. These properties can be reviewed at any time by selecting Report Properties dialog box from the Edit menu. A layout with a single section named DetailSection is created.
12. From the Tools menu, select **Set Layout Driver**. The Layout Driver dialog box opens.
13. In the Name box, enter **1*_your_initials*_Multi_Drv**.

    Because your 1*_your_initials*_Multi_Drv program does not have prompts or expect any parameters to be passed to it, leave the Parameters box blank. Your Layout Driver dialog box should look similar to the following example:



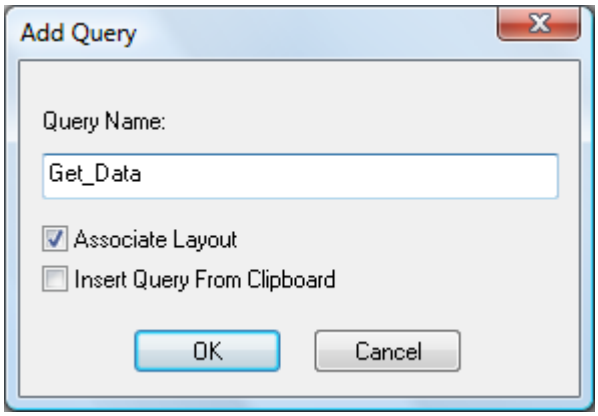

    If your 1*_your_initials*_Multi_Drv program, is a cclgroup1 object, you will need to append :group1 to the program name when you enter it in the Name: box. Use CCLPROT to determine if your program is a cclgroup1 or cclgroup0 (DBA) object.
14. Click **OK** to close the Layout Driver dialog box.
15. From the Tools menu, select **Record Builder**. The Add Records dialog box opens. Because the record structure definition already exists in the driver program, you can use Load Object to have the Record Builder pull the definition from your 1*_your_initialsMulti_Drv program into your 1your_initials* _Multi_Layout program.
16. Click **Load Object**. The Load Object dialog box opens.
17. In the Object Name box, enter **1*_your_initials*_Multi_Drv** and click **OK**. The Select Records dialog box opens.

Each record structure defined in the program object is displayed in the Select Records dialog box. Because your 1_*your_initials*_Multi_Drv program only has one record structure definition, your Select Records dialog box should look similar to the following example:

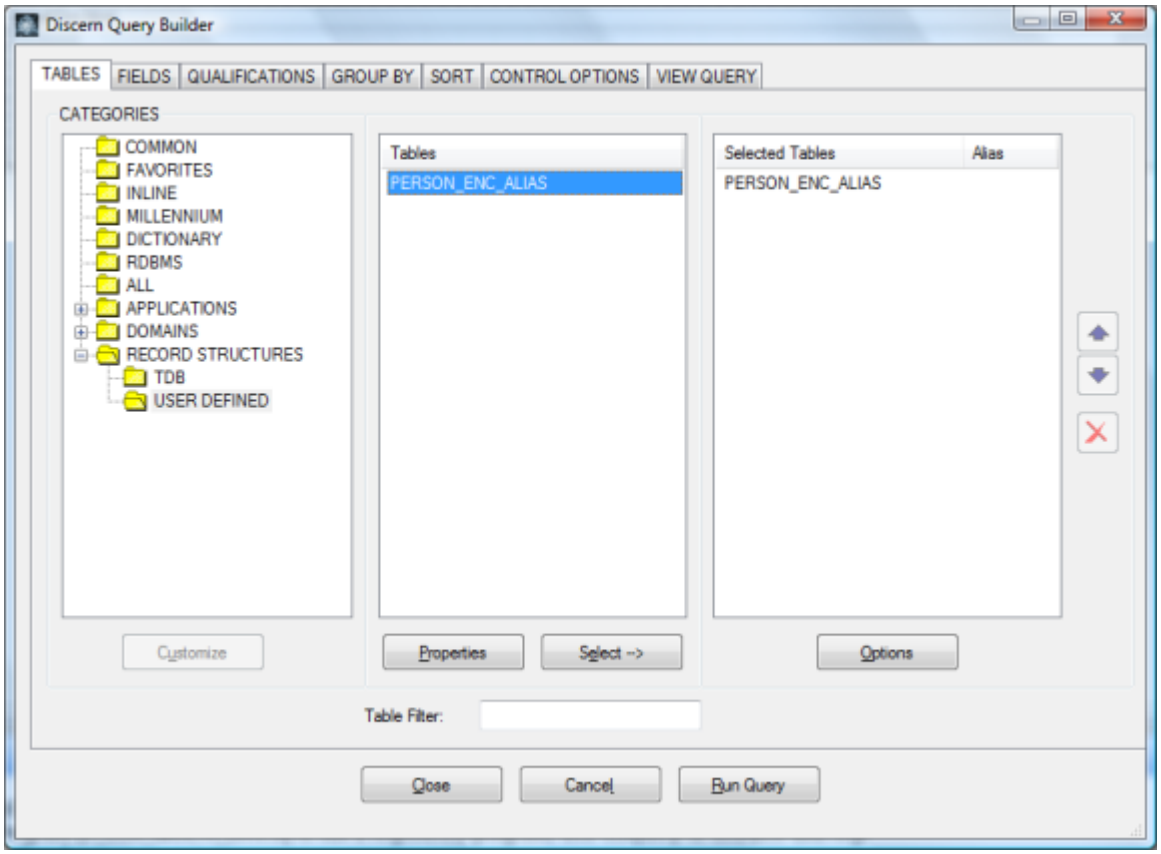



18. Select the Person_Enc_Alias record and click **OK**.
19. Click **OK** to close the Add Records dialog box.

    At this point you have the record command in the layout code and in the program. Since the layout program calls the driver program, it is not necessary to define the record in both 1_*your_initialsMulti_Layout and the 1your_initials*_MULTI_DRV programs. For the subsequent steps defined below, the record command in the driver program has been deleted. Having the record command in the driver results in the record structure listed two times in the Layout Source [Code Segment] dialog box.
20. From the Tools menu, select **Query Builder**. The Ad Queries dialog box opens.
21. Click **Add**. The Add Query dialog box opens.
22. In the Query Name box, enter **Get_Data** and verify that the Associate Layout is selected. Your Add Query dialog box should look similar to the following example:

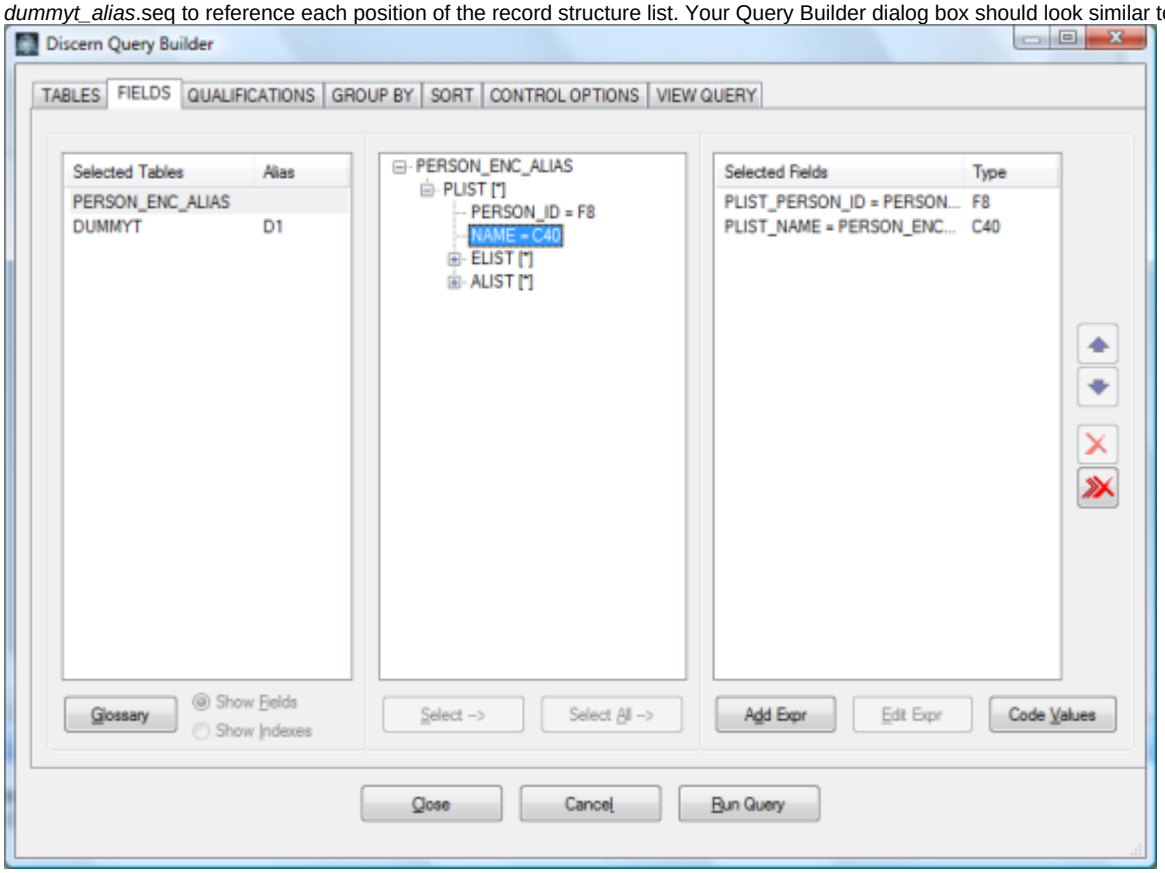



23. Click **OK**. The Discern Query Builder window opens.
24. In the **Tables** tab, expand the Record Structures list and select the User Defined category. The PERSON_ENC_ALIAS record structure is displayed in the Tables list.
25. Double-click the PERSON_ENC_ALIAS record structure to move it to the Selected Tables column. Your Discern Query Builder window should look similar to the following example:




26. In the **Fields** tab expand the PLIST and double-click the PERSON_ID and NAME items to add them to the Selected Fields list.

    Double-clicking a record structure list item adds the DUMMYT table to the **Tables** tab and creates expressions in the Selected Fields column using *dummyt_alias*.seq to reference each position of the record structure list. Your Query Builder dialog box should look similar to the following example:

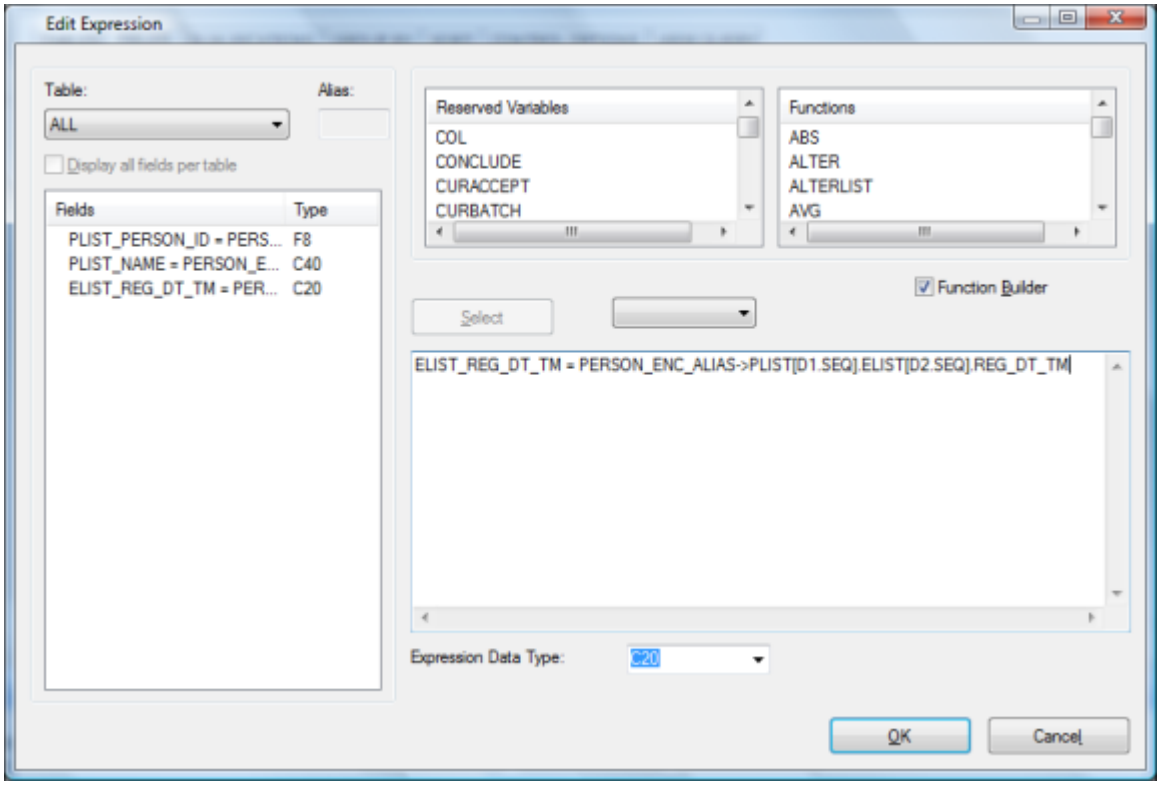



27. In the **Fields** tab, expand the ELIST and double-click the Reg_DT_TM item to add it to the Selected fields list.

    Since the REG_DT_TM item is under the Level 2 ELIST, the Query Builder adds a second reference to the DUMMYT table to the **Tables** tab and

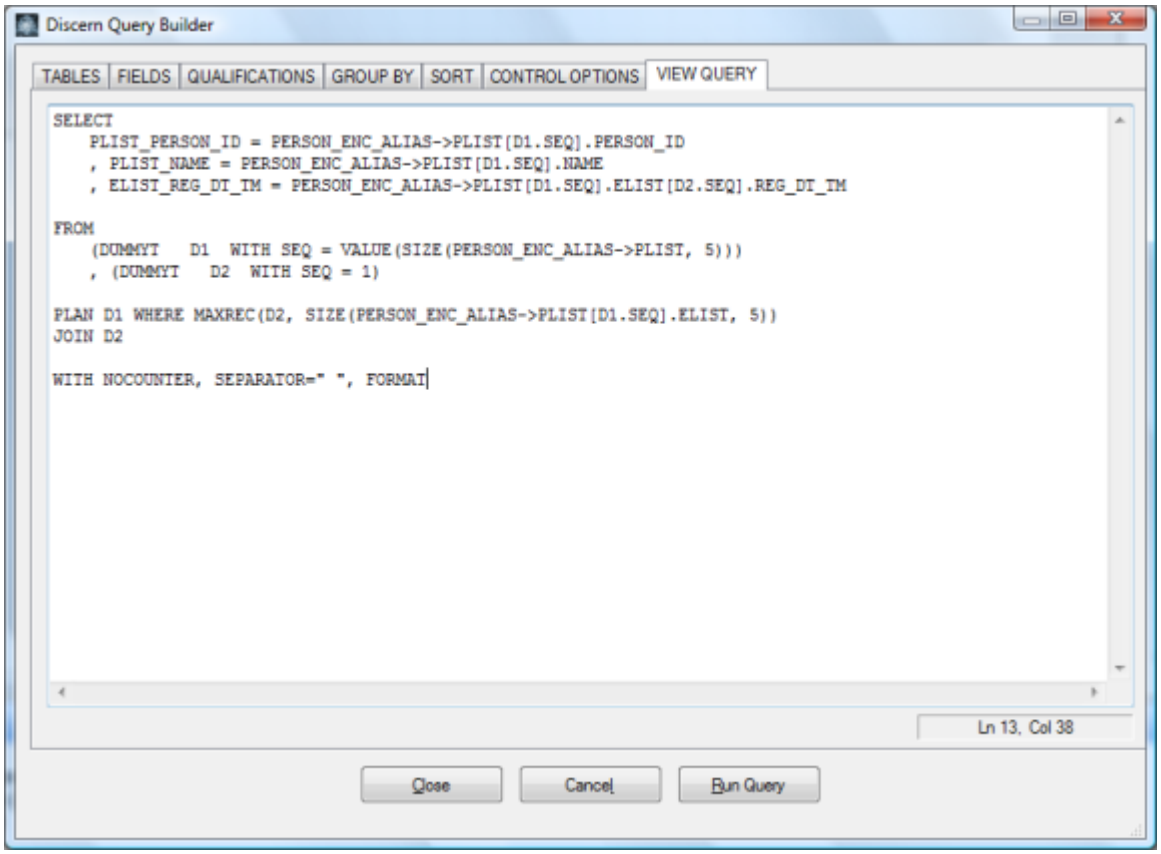
creates an expression in the selected fields column using *dummyt_alias*.seq to reference each position of the list.

28. Right-click the ELIST_REG_DT_TM expression that was created when you double-clicked the REG_DT_TM item and select **Edit Expression** to open the Edit Expression dialog shown below.




Notice in the ELIST_REG_DT_TM expression created above that the Query Builder is using D1.SEQ to reference the different positions of the PLIST and D2.SEQ to reference the different positions of the ELIST.

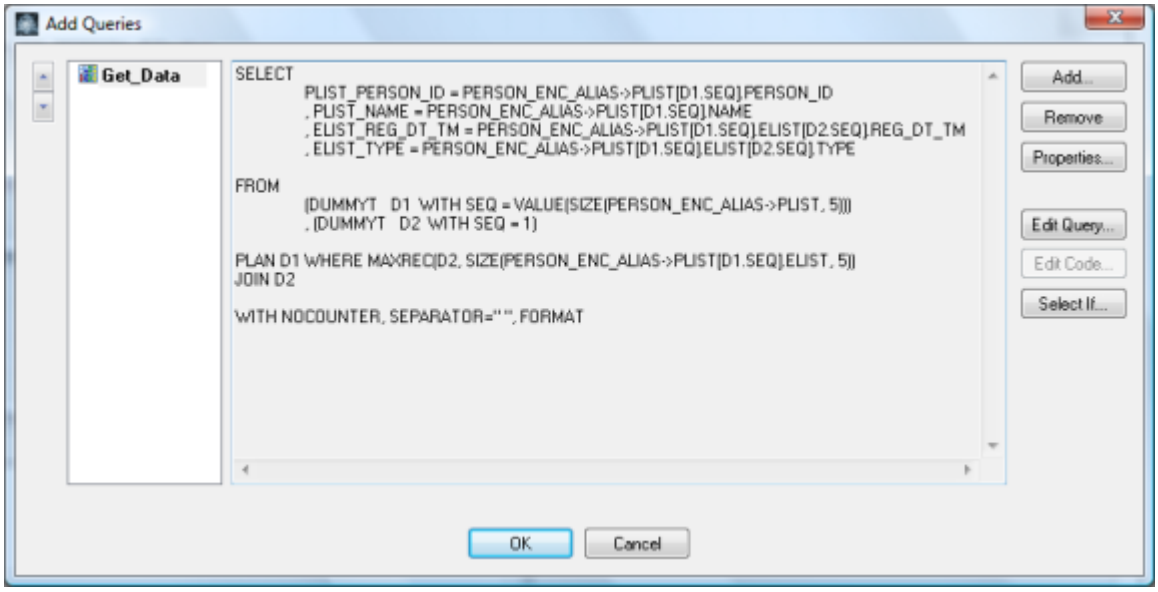
29. Click **Cancel** to close the Edit Expression dialog box.
30. Click the **View Query** tab. Your Query Builder is displayed similar to the following:




Notice that the Query Builder is defining the first reference to the DUMMYT table as D1 with Seq = the size of the PLIST. This causes D1.Seq to be set

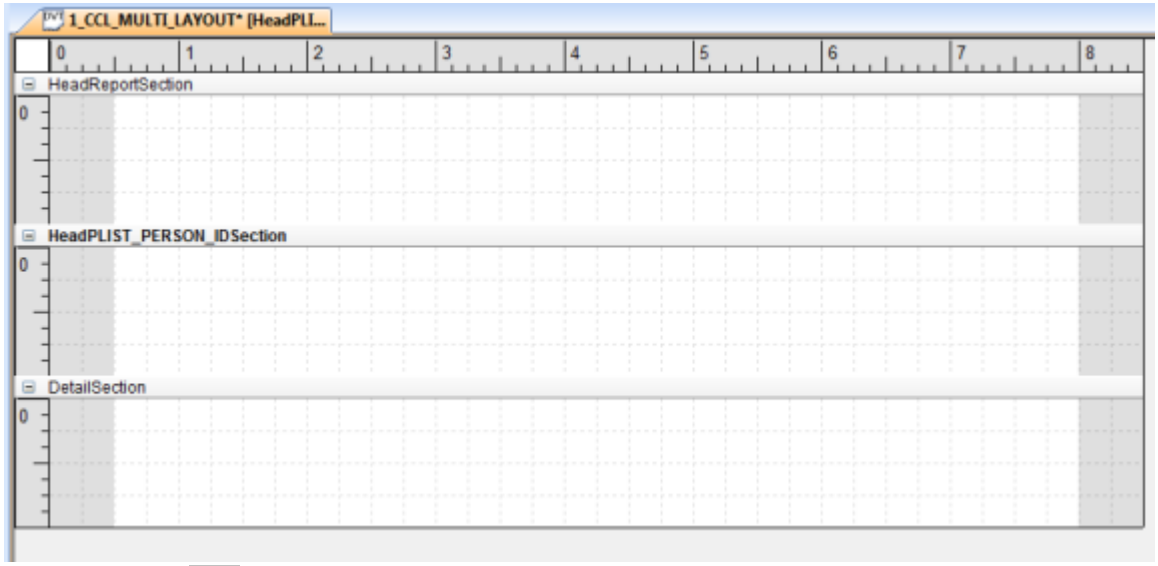
to 1 and incremented by 1 until it reaches the total number of positions in PLIST. This allows D1.Seq to be used to look at each position of the PLIST. The second reference to the DUMMYT table sets Seq = 1. In the Plan clause, the MaxRec() function is used get the size of each ELIST. Using the MaxRec() function causes D2.Seq to be set to 1 and incremented by one until it reaches the total number of positions in each of the ELISTs. This allows Discern Explorer to use D1.seq and D2.Seq in an internal looping structure to traverse the Level 2 Elists contained within the Level 1 PLIST.

31. Click the **Fields** tab.
32. From the ELIST tree, select Type to add it to the Selected Fields list.
33. Click the **View Query** tab. Your Query Builder should look similar to the following example:



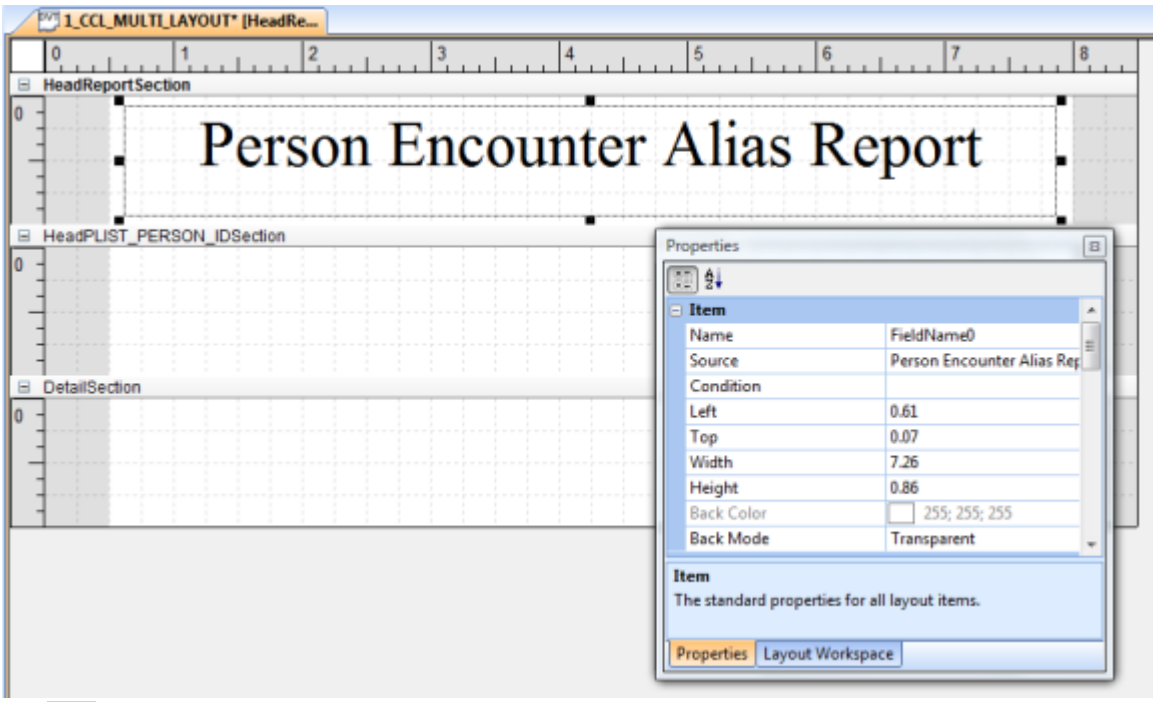

Creating expressions for the PERSON_ID and NAME items in the PLIST and for the REG_DT_TM and TYPE items in the ELIST makes it easier to display these items in your layout.

34. Click the **Sort** tab and sort by the PLIST_PERSON_ID expression.
35. Click **Close** to close the Query Builder dialog box.
36. Click **OK** to close the Add Queries dialog box.
37. From the Select/Modify Section of the Workspace dialog box, select the Head Report Reportwriter check box. Click **Yes** to create a layout section associated to the Reportwriter section.
38. Select the Head PLIST_PERSON_ID Reportwriter Sections check box and create a new Layout Section.
39. Select the Detail Reportwriter Sections check box. Since the DetailSection Layout Section already exists, you are not prompted to create it. Your layout should be similar to the following:




40. Use the Label Tool  to add a report title Person Encounter Alias Report to the HeadReportSection, and size the label box to take up most of the HeadReportSection grid.
41. Use the Formatting toolbar to set the following properties:
    1. The font to Times
    2. The font size to 36
    3. Center the text

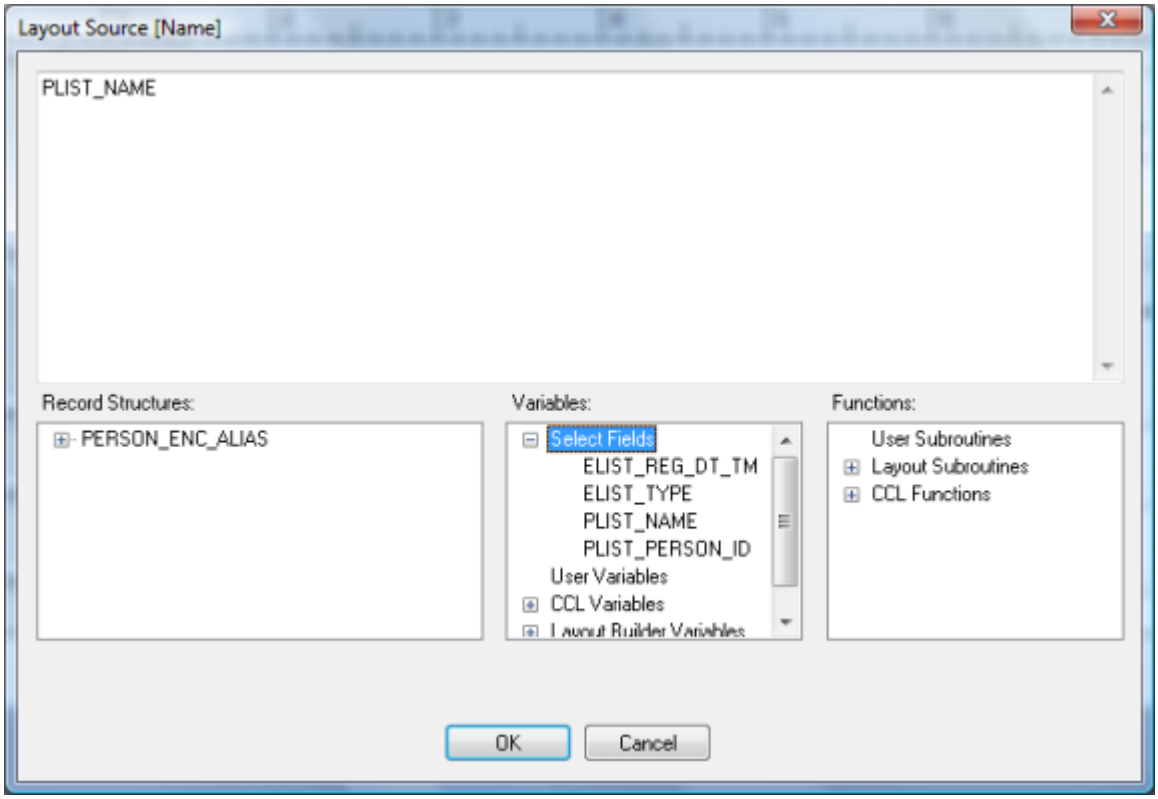
    Your layout should resemble the following example:




42. Use the Line  Tool to draw a line across the top of the HeadPLIST_PERSON_IDSection.
43. In the Properties dialog box, set the Pen Size to 0.028.

44. Use the Text Tool  to add a field to the left side of the HeadPLIST_PERSON_IDSection to display each name from the record structure.
45. In the Properties dialog box, modify the Name property to **Name**.

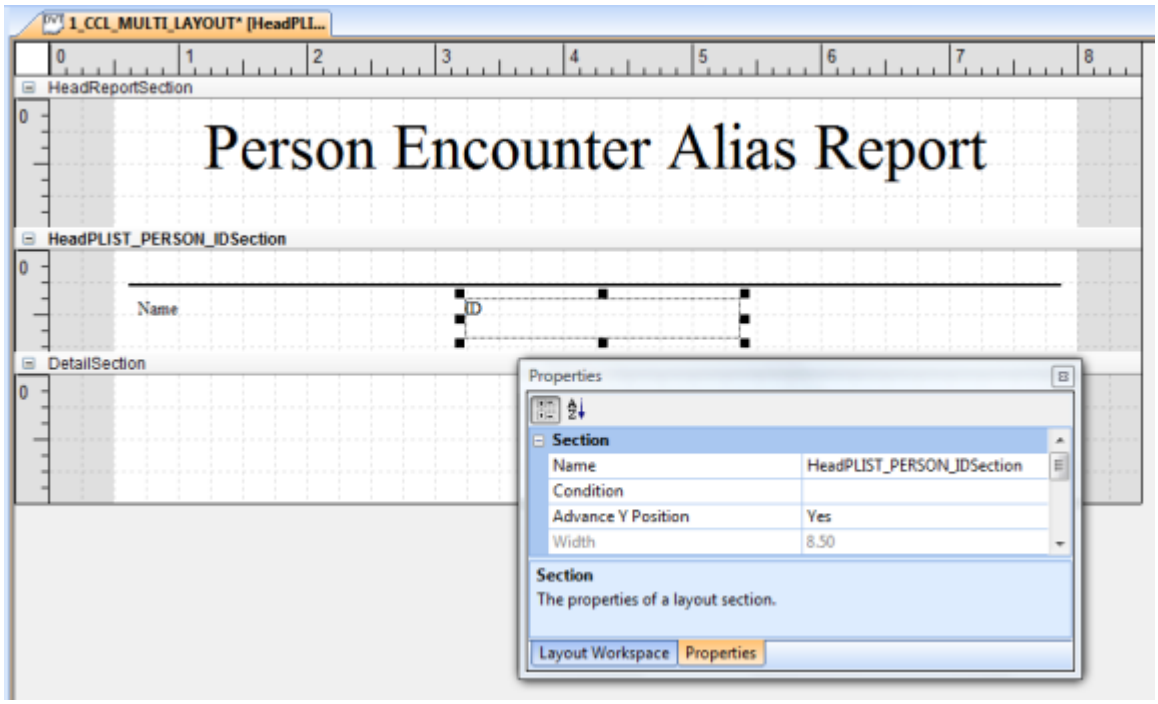
46. In the Source box click the ellipsis  button. The Layout Source [Name] dialog box opens.
47. Expand the Select Fields list in the Variables: column and double-click the PLIST_NAME expression to add it as the Layout Source. Your Layout Source [Name] dialog box should look similar to the following example:



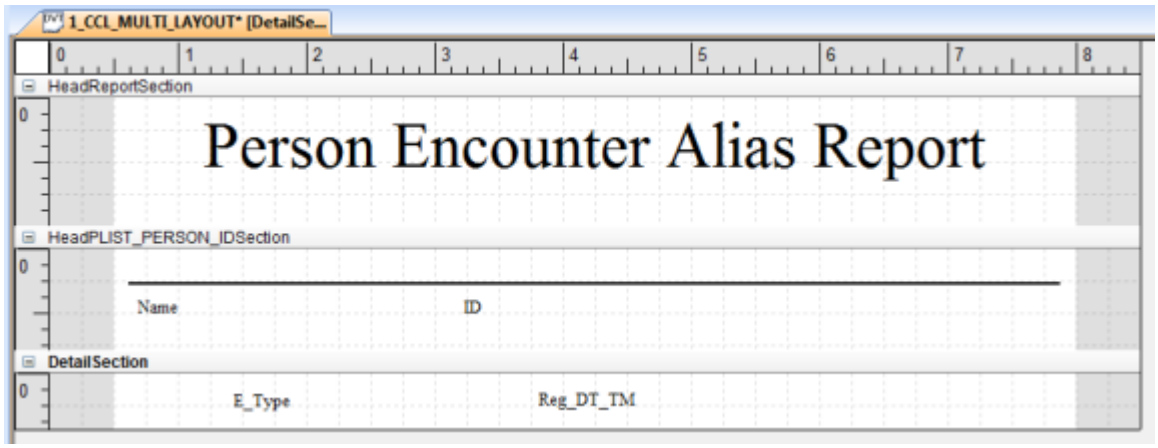

48. Click **OK** to close the Layout Source [Name] dialog box.

49. Use the Text Tool  to add a field towards the middle of the HeadPLIST_PERSON_IDSection to display the Person_ID from the record structure.
50. In the Properties dialog box, modify the Name property to **ID**.

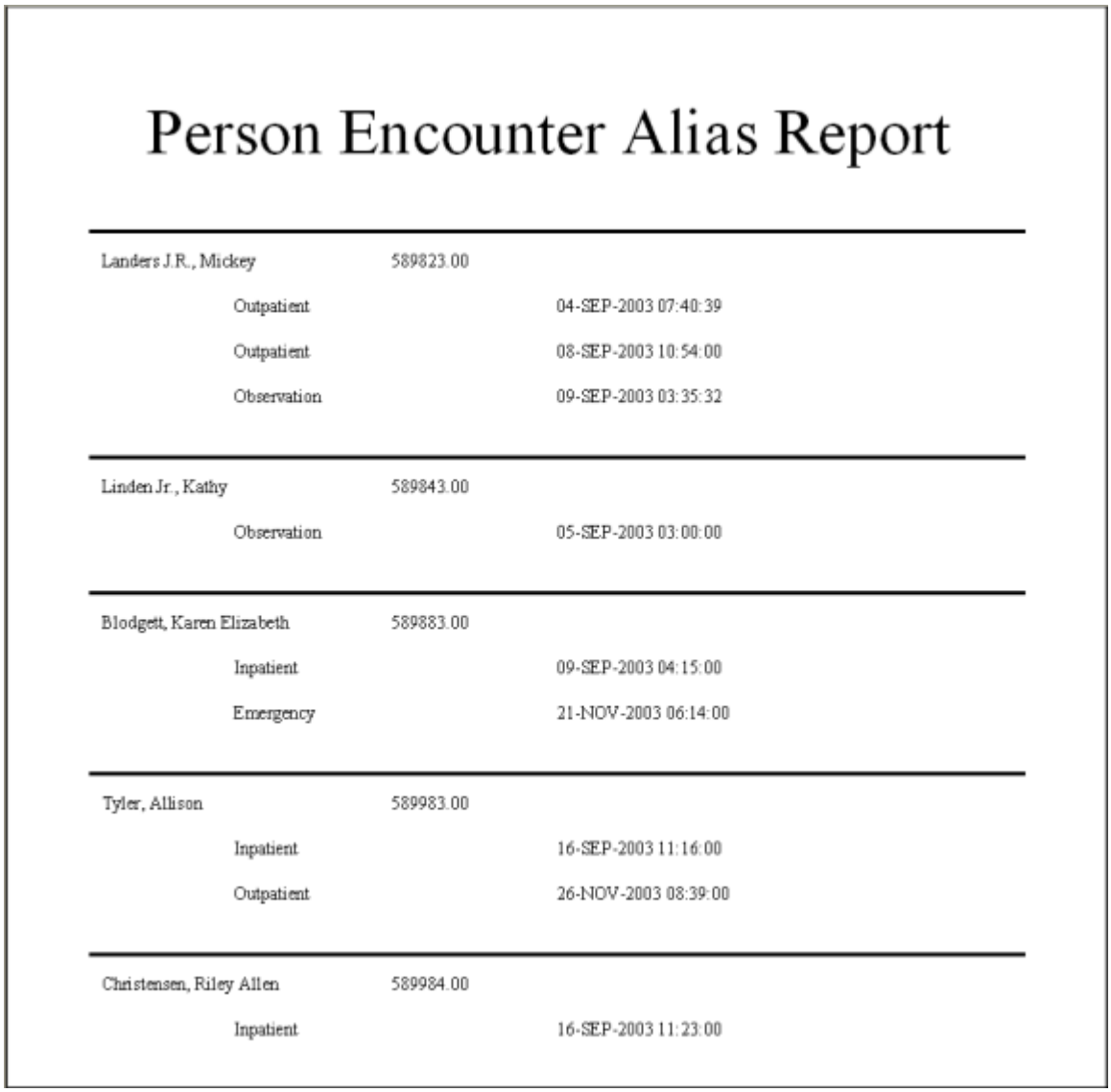
51. In the Source box click the ellipsis  button. The Layout Source [ID] dialog box opens.
52. Expand the Select Fields list in the Variables: column and double-click the PLIST_PERSON_ID expression to add it as the Layout Source.
53. Click **OK** to close the Layout Source [ID] dialog box.
54. Slowly move your pointer over the DetailSection title bar. At the top of the title bar, the pointer changes to the vertical resize pointer. Use the vertical resize to move the bottom of the HeadPLIST_PERSON_IDSection up to just underneath the items you just added. Your layout should look similar to the following example:




55. Use the Text Tool  to add a field towards the right side of the DetailSection to display the encounter type from the record structure.
56. In Properties dialog box, modify the Name property to **E_Type**.
57. In the Source box, click the ellipsis  button. The Layout Source [E_Type] dialog box opens.
58. Expand the Select Fields list in the Variables: column and double-click the ELIST_TYPE expression to add it as the Layout Source.
59. Click **OK** to close the Layout Source [E_Type] dialog box.
60. Use the Text Tool  to add a field towards the middle of the DetailSection to display the registration date and time from the record structure.
61. In the Properties dialog box, modify the Name property to **Reg_DT_TM**.
62. In the Source box, click the ellipsis  button. The Layout Source [Reg_DT_TM] dialog box opens.
63. Expand the Select Fields list in the Variables column and double-click the Elist_Reg_DT_TM expression to add it as the Layout Source.
64. Click **OK** to close the Layout Source [Reg_DT_TM] dialog box.
65. Use the vertical resize to move the bottom of the DetailSection up to just underneath the items you just added. Your layout should look similar to the following example:




66. Save the layout.
67. Execute the layout by selecting Run "1_*your_initials*_Multi_Layout" from the Build menu or clicking **Run Prompt Program** on the toolbar. Your output should look similar to the following example:

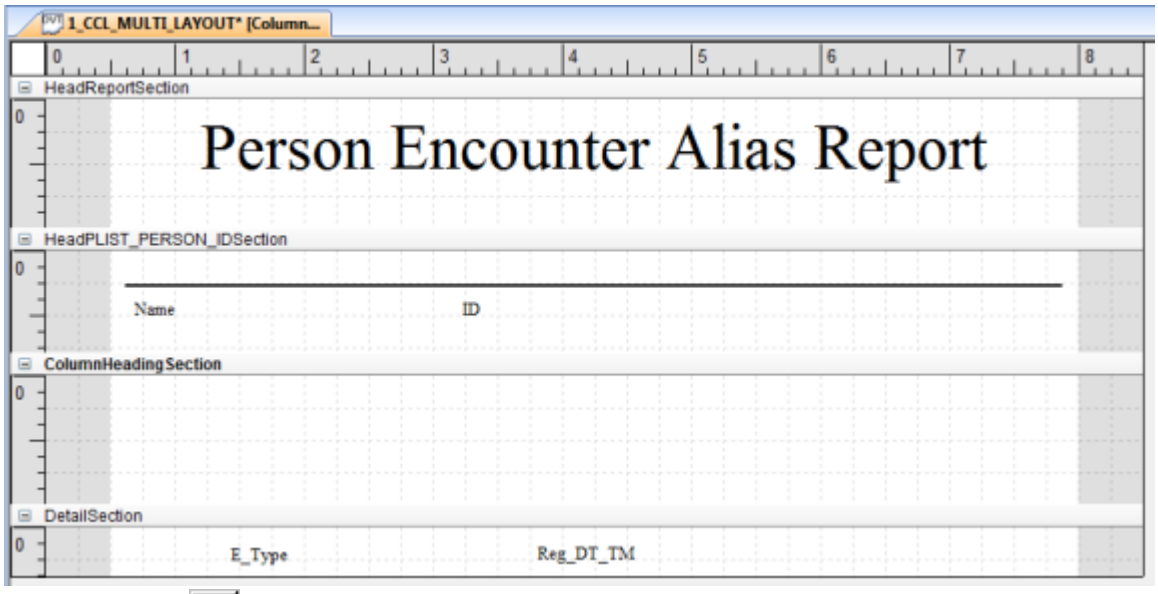


# Person Encounter Alias Report

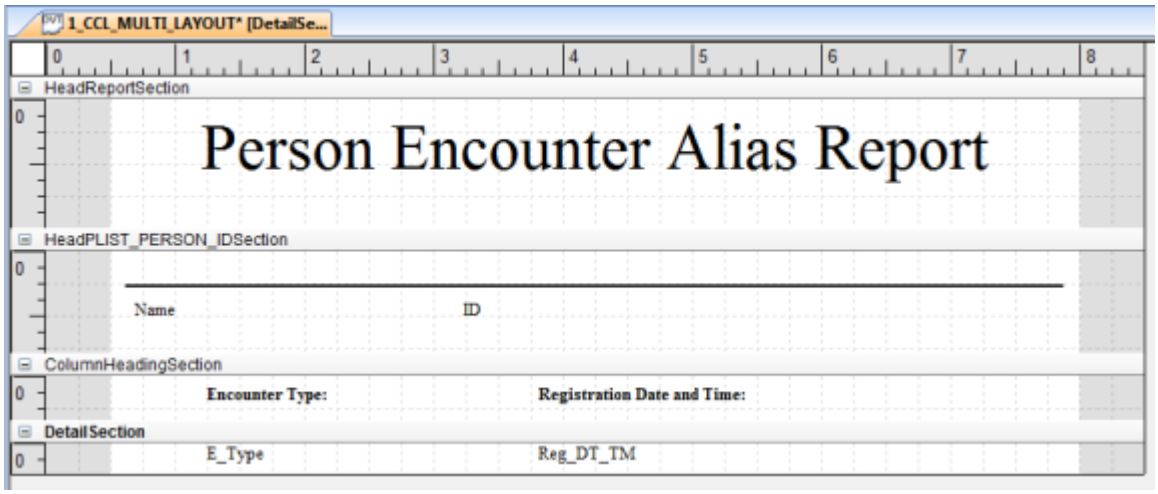
| | | |
|---|---|---|
| Landers J.R., Mickey | 589823.00 | |
| Outpatient | | 04-SEP-2003 07:40:39 |
| Outpatient | | 08-SEP-2003 10:54:00 |
| Observation | | 09-SEP-2003 03:35:32 |
| Linden Jr., Kathy | 589843.00 | |
| Observation | | 05-SEP-2003 03:00:00 |
| Blodgett, Karen Elizabeth | 589883.00 | |
| Inpatient | | 09-SEP-2003 04:15:00 |
| Emergency | | 21-NOV-2003 06:14:00 |
| Tyler, Allison | 589983.00 | |
| Inpatient | | 16-SEP-2003 11:16:00 |
| Outpatient | | 26-NOV-2003 08:39:00 |
| Christensen, Riley Allen | 589984.00 | |
| Inpatient | | 16-SEP-2003 11:23:00 |

68. Close the output display.

    The appearance of the report would be enhanced if column headings were displayed above the encounter type and the registration date and time.
69. Click the HeadPLIST_PERSON_IDSection to ensure it is active.
70. From the Select/Modify Section of the Workspace dialog box, select the Head PLIST_PERSON_ID Reportwriter section (click the label, not the check box).
71. Click **New Layout Section**  . A new layout section called HeadPLIST_PERSON_IDSection1 is created and associated to the Head PLIST_PERSON_ID reportwriter section.
72. In the Properties dialog box, modify the name of the new section to **ColumnHeadingSection**. Your layout should be similar to the following:

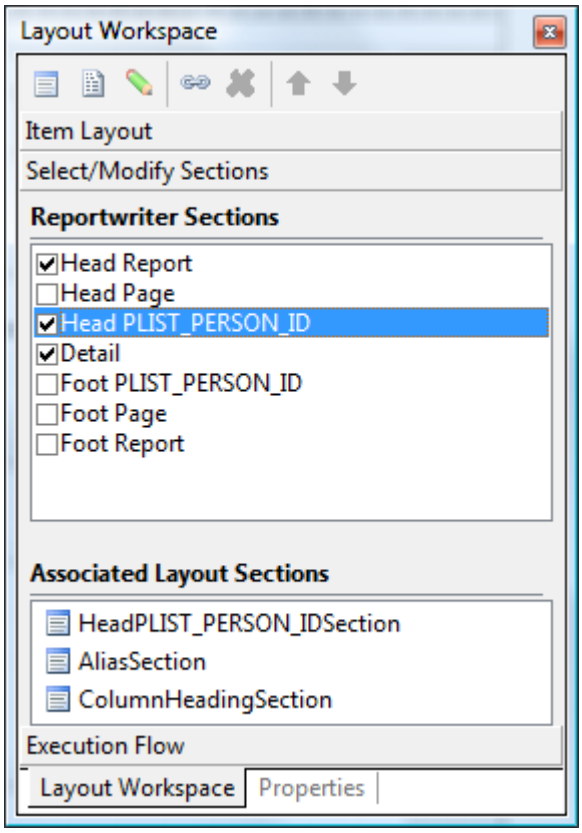



73. Use the Label Tool  to add Encounter Type: to the Source property to use as a column heading.
74. Use the Label Tool  to add Registration Date and Time: to the Source property to use as a column heading.
75. Use the formatting toolbar to bold both of the labels you have added.
76. Use the vertical resize to move the bottom of the DetailSection up to just underneath the items you just added. Your layout should look similar to the following example:

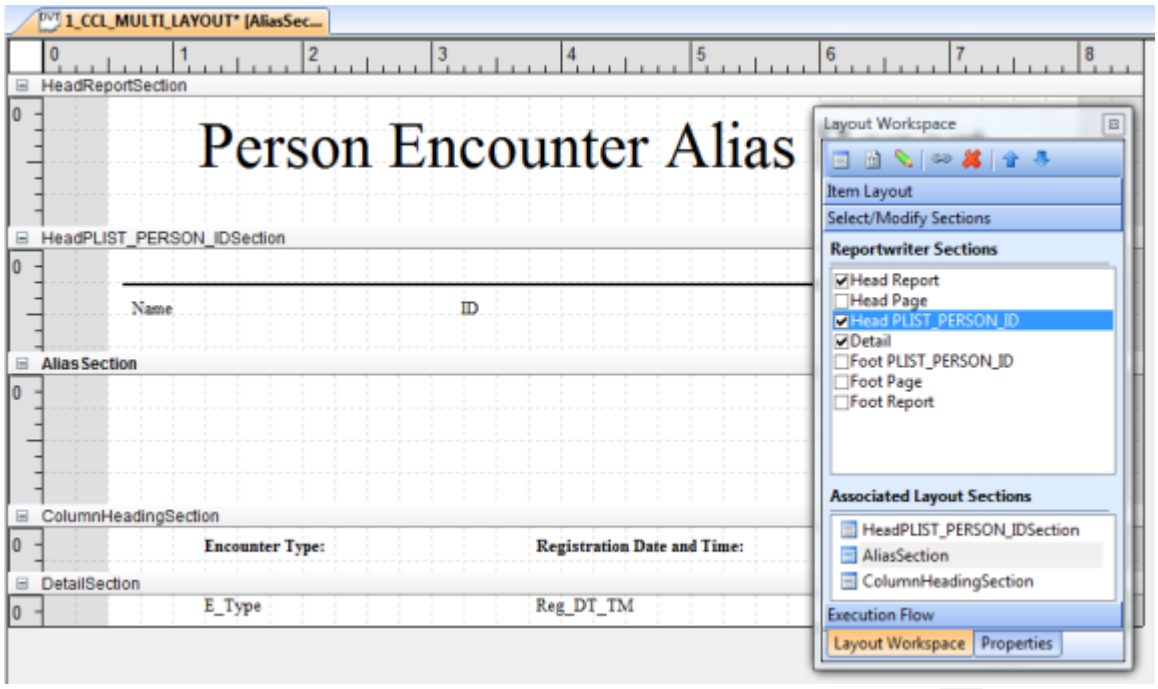



You are now ready to display the aliases for each person. The Get_Data query that you built using the Query Builder creates select expressions using an internal Discern Explorer process to traverse the Level 1 PLIST and the Level 2 ELIST. We need to create a process that traverses the Level 2 ALIST each time we get a new person and display that person's aliases on the layout. One way to accomplish this is to add a layout section that is associated with the Head PLIST_PERSON_ID reportwriter section. Code segments can be used to determine the number of aliases that exist in the ALIST for the person and execute a For Loop to traverse the ALIST and call the layout section to display all of the person's aliases.

77. Click the HeadPLIST_PERSON_IDSection to ensure it is active.
78. From the Select/Modify Section of the Workspace dialog box, select the Head PLIST_PERSON_ID Reportwriter section (click the label, not the check box)
79. Click the **New Layout Section** button  . A new layout section called HeadPLIST_PERSON_IDSection1 is created and associated to the Head PLIST_PERSON_ID reportwriter section.
80. In the Properties dialog box, modify the name of the new section to **AliasSection**.
81. From the Select/Modify Section of the Workspace dialog box, select the Head PLIST_PERSON_ID Reportwriter section (click the label, not the check box).
82. Select the AliasSection from the Associated Layout Section and click the **Move up** button  to place the section in the middle. Your Associated Layout Section should be displayed similar to the following example:



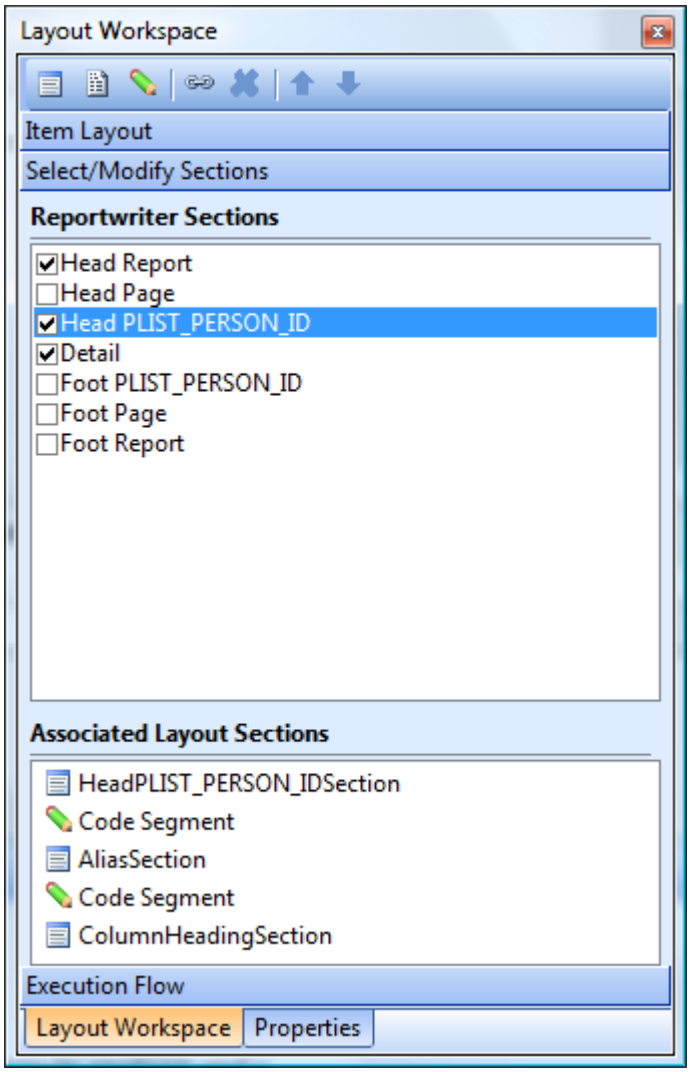

83. Move the AliasSection so that visually it is between the HeadPLIST_PERSON_ID and the ColumnHeadingSection. This can be done by placing your cursor on the ruler for the AliasSection, and then dragging it to the middle of the ruler for the ColumnHeadingSection. Your layout should look similar to the following example:




84. With the Head PLIST_PERSON_ID reportwriter section still selected, click the New Code Segment button on the Layout Workspace toolbar twice. Two new code segments are inserted in to the Reportwriter Section.

85. Select the first Code Segment and place it underneath the HeadPLIST_PERSON_IDSection using the **Move Up** button .

86. Select the second Code Segment and place it underneath the AliasSection using the Move Up button . Your Associated Layout Section for the Head PLIST_PERSON_ID reportwriter section should resemble the following:



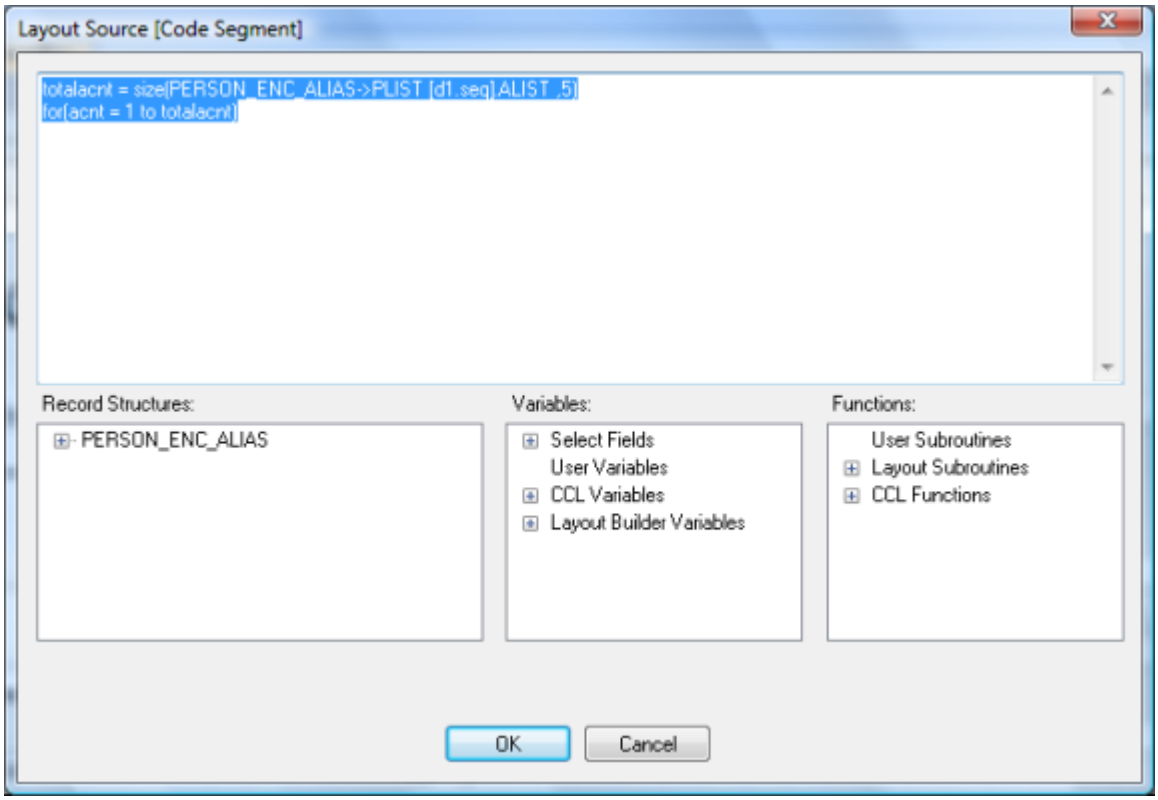

Adding the layout sections and code segments to the Head PLIST_PERSON_ID reportwriter section causes each layout section and code segment to be executed in the order they appear in the Associated Layout Sections list. This is done each time a new person ID is encountered in the result set of the query that is associated with the layout.

87. Double-click the first Code Segment section you added above. The Layout Source [Code Segment] dialog box opens.
88. Enter the following command to set a variable named TotalAcnt equal to the number of items in the ALIST for each person in the PLIST:

    You can enter the entire command, or build it by using a combination of typing and selecting the Size function from the CCL Functions list in the Functions column, expanding the Person_Enc_Alias record structure in the Record Structures: column, and then double-clicking the ALIST.

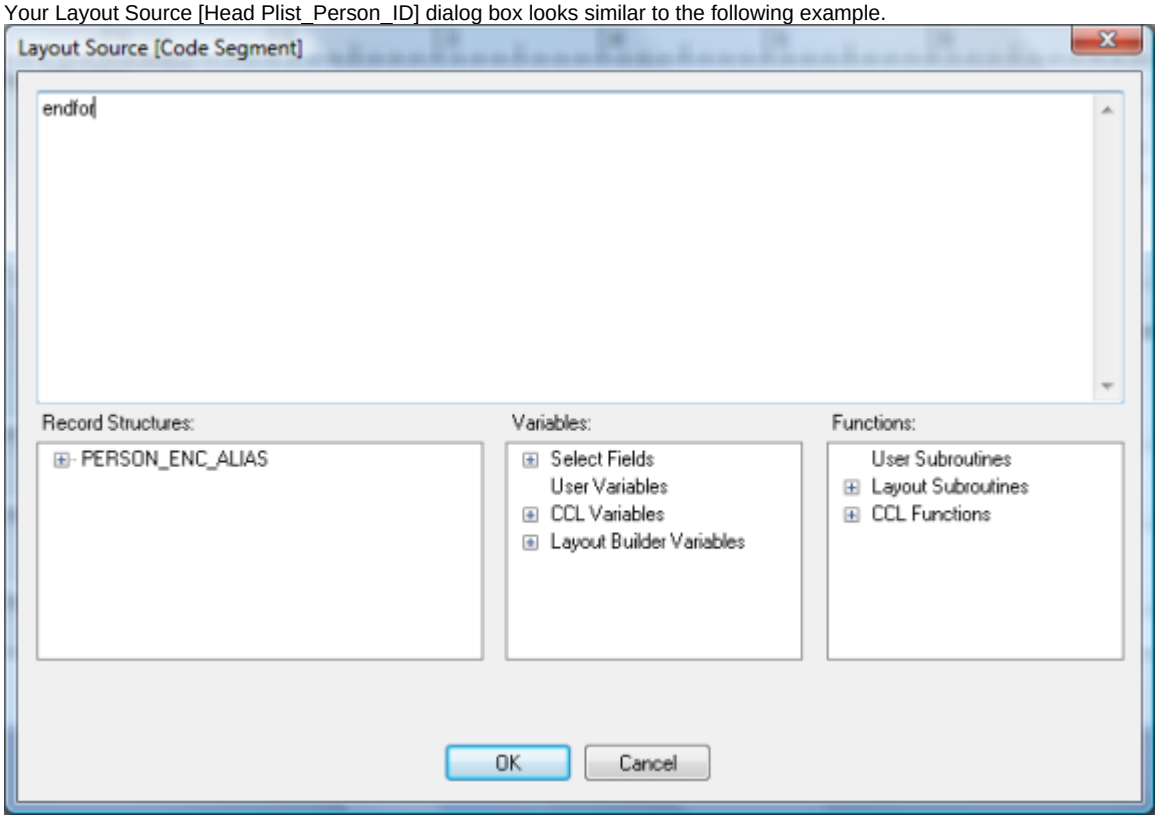
89. Enter the following command to start a For Loop that traverses all of the aliases in the ALIST for the current person:

    Your Layout Source [Code Segment] dialog box should look similar to the following example:




90. Click **OK** to close the Layout Source [Code Segment] dialog.
91. Double-click the second Code Segment section you added above. The Layout Source [Code Segment] dialog box opens.
92. Enter the following code to terminate the For Loop:

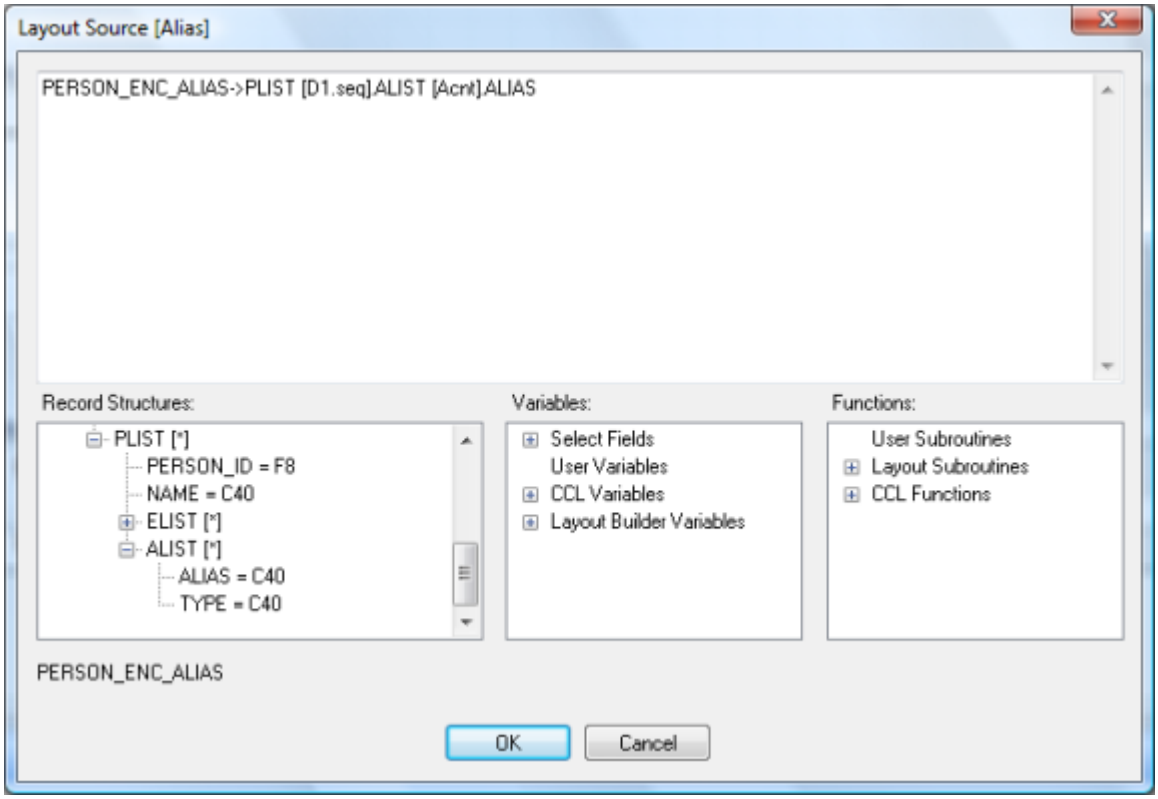
Your Layout Source [Head Plist_Person_ID] dialog box looks similar to the following example.




93. Click **OK** to close the Layout Source [Code Segment] dialog box.
94. Use the Text Tool  to add a field to the AliasSection to display each alias from the record structure.
95. In the Properties dialog box, modify the Name property to **Alias**.

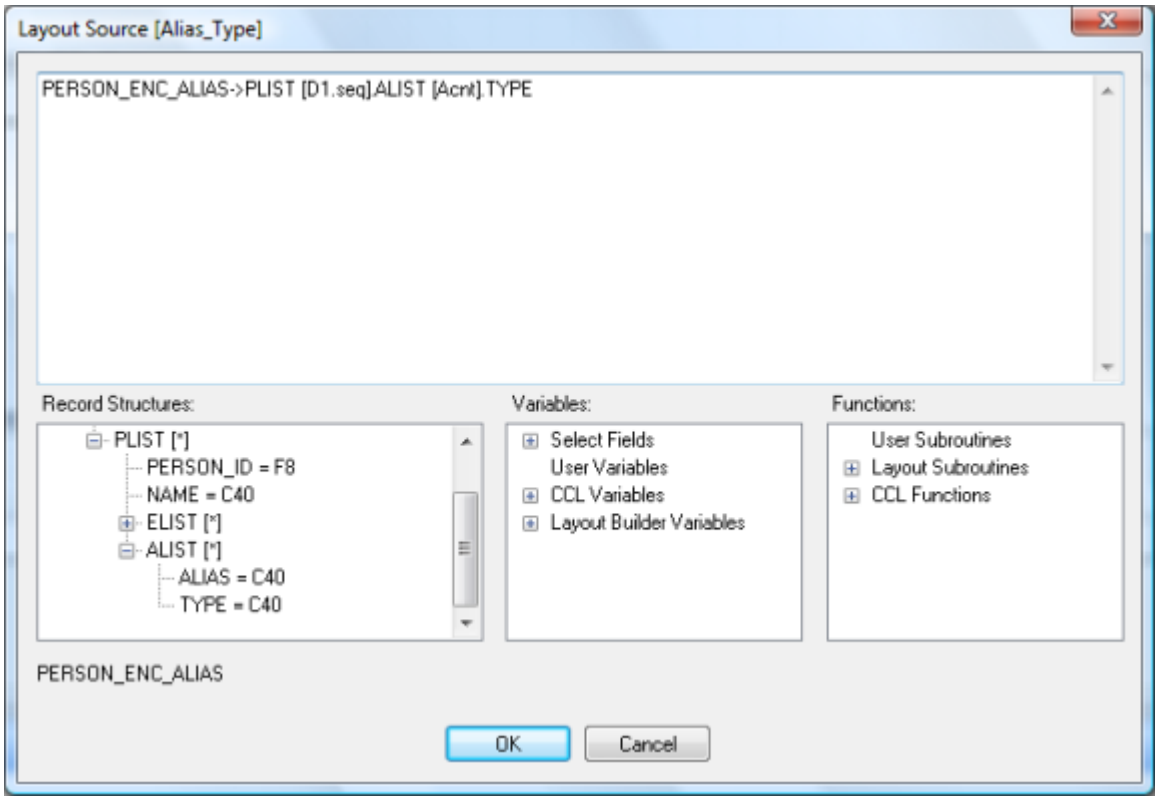
96. In the Source box, click the ellipsis  button to open the Layout Source [Alias] dialog box.
97. In the Record Structures list, expand the PERSON_ENC_ALIAS tree.
98. Expand the PLIST [*] tree.
99. Expand the ALIST [*] tree.
100. Double-click Alias = C40 to add the following code as the layout source:

101. Enter **D1.seq** as the subscript for the PLIST.
102. Enter **Acnt** as the subscript for the ALIST. You Layout Source now should look similar to the following example:

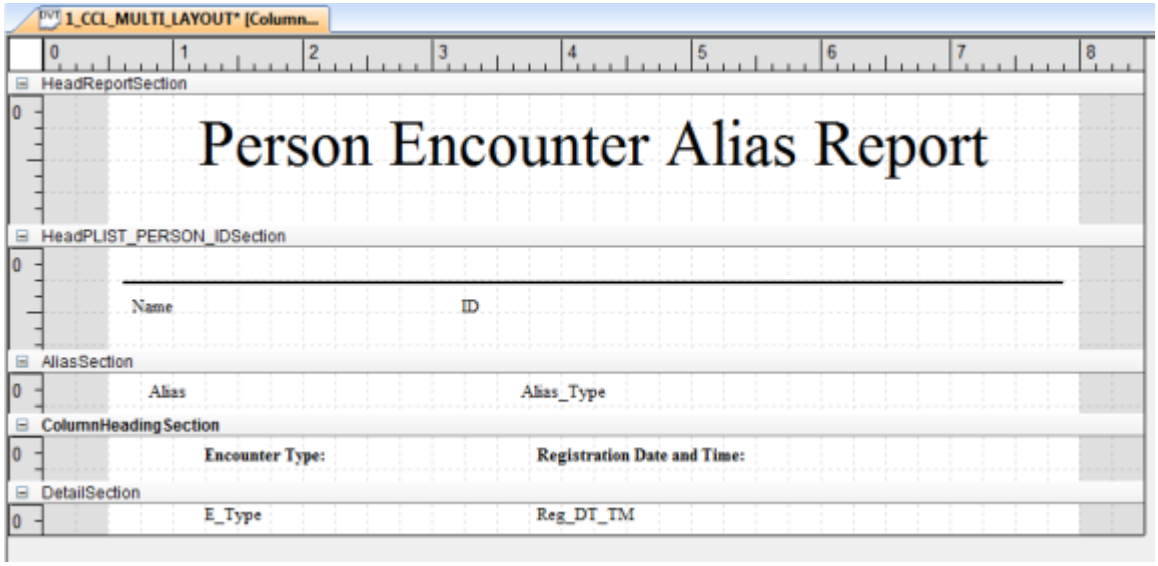



103. Click **OK** to close the Layout Source [Alias] dialog box.
104. Use the Text Tool  to add a field to the AliasSection to display each alias type from the record structure.
105. In the Properties dialog box, modify the Name property to **Alias_Type**.
106. In the Source box, click the ellipsis  button to open the Layout Source [Alias_Type] dialog box.
107. In the Record Structures list, expand the PERSON_ENC_ALIAS tree.
108. Expand the PLIST [*] tree.
109. Expand the ALIST [*] tree.
110. Double-click Type = C40 to add the following code as the layout source:

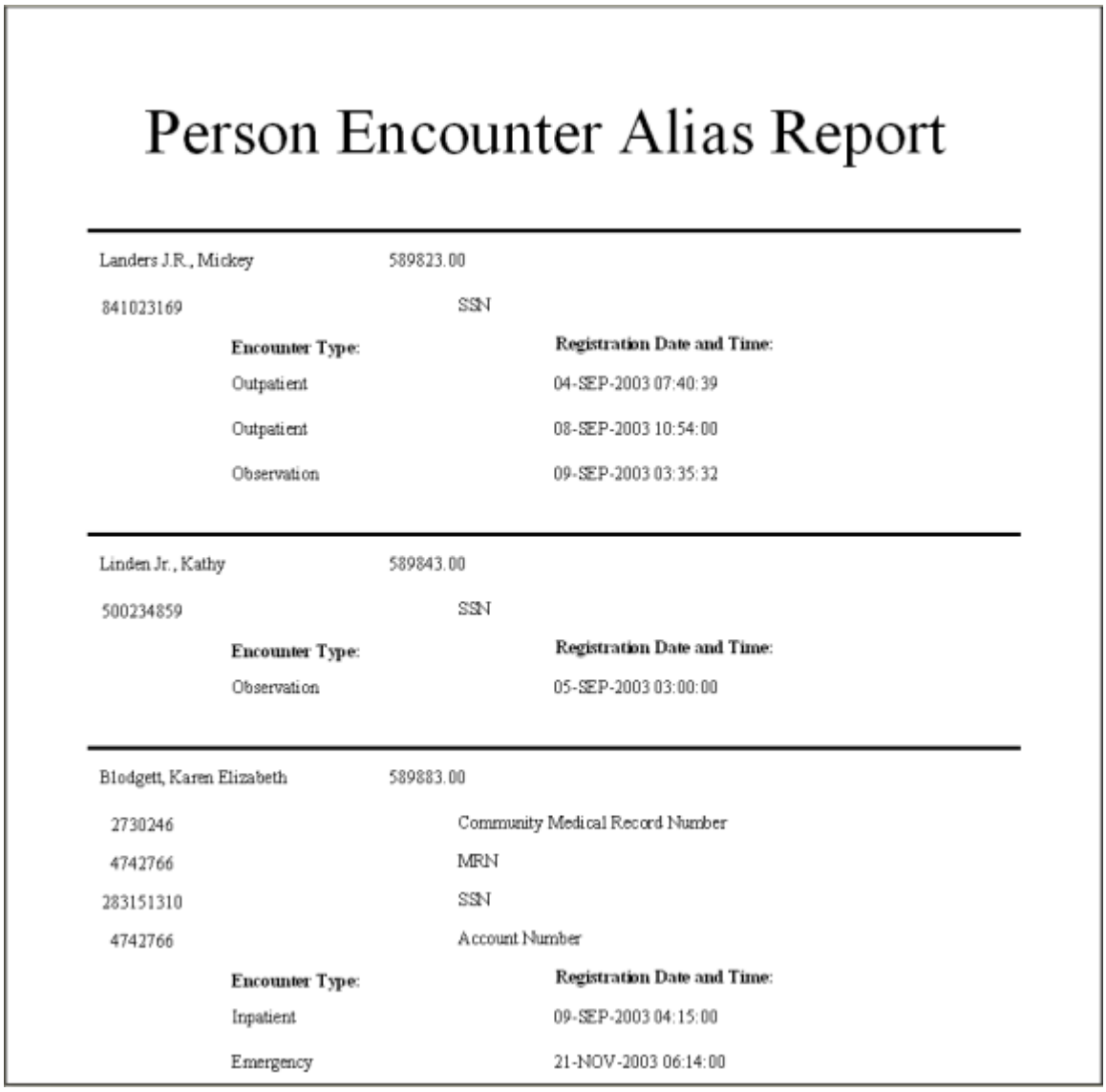
111. Enter **D1.seq** as the subscript for the PLIST.
112. Enter **Acnt** as the subscript for the ALIST. Your Layout Source now should look similar to the following example:




113. Click **OK** to close the Layout Source dialog box.
114. Use the vertical resize to move the bottom of the Column/HeadingSection up to just underneath the items you just added. Your layout should look similar to the following example:




115. Save the layout.
116. Execute the layout by selecting **Run "1_*your_initials*_Multi_Layout_Layout"** from the Build menu or clicking **Run Prompt Program** from the toolbar. Your output should look similar to the following example:



# Person Encounter Alias Report

Landers J.R., Mickey 589823.00

841023169 SSN

| Encounter Type: | Registration Date and Time: |
|---|---|
| Outpatient | 04-SEP-2003 07:40:39 |
| Outpatient | 08-SEP-2003 10:54:00 |
| Observation | 09-SEP-2003 03:35:32 |

Linden Jr., Kathy 589843.00

500234859 SSN

| Encounter Type: | Registration Date and Time: |
|---|---|
| Observation | 05-SEP-2003 03:00:00 |

Blodgett, Karen Elizabeth 589883.00

2730246 Community Medical Record Number

4742766 MRN

283151310 SSN

4742766 Account Number

| Encounter Type: | Registration Date and Time: |
|---|---|
| Inpatient | 09-SEP-2003 04:15:00 |
| Emergency | 21-NOV-2003 06:14:00 |

Once you complete these steps, continue to the next part of Use Discern Layout Builder, Sub Reports.