RMS Report Designing
RMS Report Writing

Examples for designing custom report in RMS

by RMS Support Center

RMS uses the Report Builder report writing tool to allow users to design customized Reports using RMS data. This banded report writing tool is flexible and powerful, but requires patience and skill.

This document contains many examples of reports that illustrate interesting and powerful ways to access and display RMS data using the report writer. Step through the example one by one in order to get the most out of this document.
# Table of Contents

**Foreword** 0  

**Part I Example 1 - RMS Program Type** 8  
1 Overview .................................................................................................................. 8  
2 Step 1 - Add the Report .......................................................................................... 8  
3 Step 2 - Create the Query ...................................................................................... 10  
4 Step 3 - Design the Report Header ...................................................................... 14  
5 Step 4 - Design the Detail Band ............................................................................ 18  
6 Summary .................................................................................................................. 19  

**Part II Example 2 - Standard Templates** 21  
1 Standard Templates Provided ................................................................................. 21  

**Part III Example 3 - RMS Program Types (redone)** 24  
1 Overview .................................................................................................................. 24  
2 Step 1 - Load up the Standard Portrait Template ................................................. 24  
3 Step 2 - Create the Query ...................................................................................... 25  
4 Step 3 - Add DBText controls to the Detail Band .................................................. 28  
5 Step 4 - Multi-Selecting and changing properties ................................................. 29  
6 Step 5 - Micro moving and Micro Sizing using the Keyboard and Mouse .......... 30  
7 Step 6 - Aligning Objects ...................................................................................... 31  
8 Step 7 - Size and Position the DBText Items ......................................................... 32  
9 Step 8 - Add Column Titles ................................................................................... 32  
10 Step 9 - Embellishments for the Y/N column - RAP code .................................. 33  
11 Summary ............................................................................................................... 36  

**Part IV Example 4 - Interest Payments** 38  
1 Overview .................................................................................................................. 38  
2 Step 1 - Load the Landscape Template and Set the Basic Design ....................... 38  
3 Step 2 - Create the Query ...................................................................................... 38  
4 Step 3 - Design the Detail Band ............................................................................ 42  
5 Step 4 - Add a Group ............................................................................................. 42  
6 Step 5 - Add a Summary Band ............................................................................. 44  
7 Step 6 - Set 2 pass and lookahead options ........................................................... 44  
8 Summary ............................................................................................................... 46  

**Part V Example 5 - AutoSearch** 48  
1 Overview .................................................................................................................. 48
Part VI Example 6 - Special AutoSearches (Contract and Office)  62

1 Overview  ................................................................. 62
2 Step 1 - Load the existing Report ...................................................... 62
3 Step 2 - Configure Contract AutoSearch ......................................... 63
4 Step 3 - Test the Contract AutoSearch .......................................... 65
5 Step 4 - Configure Office AutoSearch ........................................... 66
6 Step 5 - Test the Office AutoSearch ............................................. 68

Part VII Example 7 - Office Symbol Join  72

1 Overview  .................................................................................. 72
2 Step 1 - Load the existing report .................................................... 72
3 Step 2 - Join to the Office Table ................................................... 73
4 Step 3 - Add the DISPLAY_SYMBOL field ................................. 75
5 Summary  .................................................................................. 77

Part VIII Example 8 - Equipment List  79

1 Overview  .................................................................................. 79
2 Step 1 - Planning the queries and Finding the Data  ......................... 79
3 Step 2 - Load the Landscape Template ............................................ 80
4 Step 3 - Create the Query ............................................................ 81
5 Step 4 - More Query Work - SQL Expressions ............................... 85
6 Step 5 - Enable the query to allow the user to select contract(s) .......... 87
7 Step 6 - Sort the query ............................................................... 89
8 Step 7 - Add Column Titles .......................................................... 90
9 Step 8 - Add a Region for the detail Band ...................................... 91
10 Step 9 - Add the DBText Objects ................................................ 92
11 Step 10 - Add the DBMemo Object .............................................. 92
<table>
<thead>
<tr>
<th>Step</th>
<th>Task Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>12</td>
<td>Step 11 - Add Grouping by Contract</td>
<td>93</td>
</tr>
<tr>
<td>13</td>
<td>Step 12 - Design the Group Header</td>
<td>94</td>
</tr>
<tr>
<td>14</td>
<td>Summary</td>
<td>96</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td><strong>Part IX</strong> Example 9 - Contract Value History</td>
<td>98</td>
</tr>
<tr>
<td>1</td>
<td>Overview</td>
<td>98</td>
</tr>
<tr>
<td>2</td>
<td>Step 1 - Load the Standard Portrait Template and Set the basic report items</td>
<td>99</td>
</tr>
<tr>
<td>3</td>
<td>Step 2 - Create a View for the detail data</td>
<td>99</td>
</tr>
<tr>
<td>4</td>
<td>Step 3 - Create 2 Data Pipelines</td>
<td>102</td>
</tr>
<tr>
<td>5</td>
<td>Step 4 - Add a subreport for detail data</td>
<td>104</td>
</tr>
<tr>
<td>6</td>
<td>Step 5 - Add a running subtotal</td>
<td>106</td>
</tr>
<tr>
<td>7</td>
<td>Step 6 - Add a grand total Summary Band to the SubReport</td>
<td>107</td>
</tr>
<tr>
<td>8</td>
<td>Summary</td>
<td>108</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td><strong>Part X</strong> Example 10 - Office Org Chart</td>
<td>110</td>
</tr>
<tr>
<td>1</td>
<td>Overview</td>
<td>110</td>
</tr>
<tr>
<td>2</td>
<td>Step 1 - Load the Portrait Template and Set Report Title</td>
<td>111</td>
</tr>
<tr>
<td>3</td>
<td>Step 2 - About the Office Hierarchy</td>
<td>111</td>
</tr>
<tr>
<td>4</td>
<td>Step 3 - Create the query</td>
<td>112</td>
</tr>
<tr>
<td>5</td>
<td>Step 4 - Create a Calculated Field for the Office Level</td>
<td>115</td>
</tr>
<tr>
<td>6</td>
<td>Step 5 - Layout the Detail Band</td>
<td>116</td>
</tr>
<tr>
<td>7</td>
<td>Step 6 - Dynamically indent the Region</td>
<td>117</td>
</tr>
<tr>
<td>8</td>
<td>Step 7 - Anchor the DBText Objects</td>
<td>117</td>
</tr>
<tr>
<td>9</td>
<td>Summary</td>
<td>120</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td><strong>Part XI</strong> Example 11 - Progress Payment Processing</td>
<td>122</td>
</tr>
<tr>
<td>1</td>
<td>Overview</td>
<td>122</td>
</tr>
<tr>
<td>2</td>
<td>Step 1 - Load the Standard Landscape Template and Design the Header</td>
<td>122</td>
</tr>
<tr>
<td>3</td>
<td>Step 2 - Create a Left Outer Join DataView</td>
<td>123</td>
</tr>
<tr>
<td>4</td>
<td>Step 3 - Filter the Contract Rows</td>
<td>126</td>
</tr>
<tr>
<td>5</td>
<td>Step 4 - Filter the Request Table Rows</td>
<td>129</td>
</tr>
<tr>
<td>6</td>
<td>Step 5 - Select the fields and the sort order</td>
<td>129</td>
</tr>
<tr>
<td>7</td>
<td>Step 6 - Create a second query for the Receipt table</td>
<td>130</td>
</tr>
<tr>
<td>8</td>
<td>Step 7 - Create Calc Fields for the Request Pipeline</td>
<td>131</td>
</tr>
<tr>
<td>9</td>
<td>Step 8 - Check the Grouping of the Query</td>
<td>132</td>
</tr>
<tr>
<td>10</td>
<td>Step 9 - Linking the DataViews</td>
<td>133</td>
</tr>
<tr>
<td>11</td>
<td>Step 10 - Add a region and DBText Field to the detail band</td>
<td>135</td>
</tr>
<tr>
<td>12</td>
<td>Step 11 - Add a Variable Object for the Invoice Number</td>
<td>136</td>
</tr>
<tr>
<td>13</td>
<td>Step 12 - Set the name of the Report Objects using the Report Tree</td>
<td>136</td>
</tr>
<tr>
<td>14</td>
<td>Step 13 - Add remaining variable objects</td>
<td>138</td>
</tr>
</tbody>
</table>
Step 15 - Add RAP code for remaining variables .............................................................. 140

Part XII Example 12 - Scheduled Construction Events 143

1 Overview .......................................................................................................................... 143

Part XIII Example 13 - Interactive Invoice Drilldown 168

1 Overview .......................................................................................................................... 168

Part XIV Example 14 - Payment Obligation Crosstab 191

1 Overview .......................................................................................................................... 191
# Contents

<table>
<thead>
<tr>
<th>Part XV</th>
<th>Example 15 - Office Workload by Fund Type (Piechart)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Overview .....................................................................</td>
</tr>
<tr>
<td>2</td>
<td>Step 1 - Load the Landscape Template ................................</td>
</tr>
<tr>
<td>3</td>
<td>Step 2 - Create the Office Query ..................................</td>
</tr>
<tr>
<td>4</td>
<td>Step 3 - Create the Contract Query .................................</td>
</tr>
<tr>
<td>5</td>
<td>Step 4 - Link the Contract Query ....................................</td>
</tr>
<tr>
<td>6</td>
<td>Step 5 - Design the main report ....................................</td>
</tr>
<tr>
<td>7</td>
<td>Step 6 - Design the subreport and chart ...........................</td>
</tr>
<tr>
<td>8</td>
<td>Step 7 - Format the Chart .............................................</td>
</tr>
<tr>
<td>9</td>
<td>Summary ........................................................................</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Part XVI</th>
<th>Example 16 - Project Data Sheet with Image</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Overview ..................................................</td>
</tr>
<tr>
<td>2</td>
<td>Step 1 - Create the Contract Query ..................</td>
</tr>
<tr>
<td>3</td>
<td>Step 2 - Create the Office Query .....................</td>
</tr>
<tr>
<td>4</td>
<td>Step 3 - Design the Report ...............................</td>
</tr>
<tr>
<td>5</td>
<td>Summary ......................................................</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Part XVII</th>
<th>Example 17 - Contractor Evaluation Grades</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Overview ................................................</td>
</tr>
<tr>
<td>2</td>
<td>Step 1 - Load the Portrait Template ..........</td>
</tr>
<tr>
<td>3</td>
<td>Step 2 - Create a View of DUNS numbers ........</td>
</tr>
<tr>
<td>4</td>
<td>Step 3 - Create the DUNS Query ....................</td>
</tr>
<tr>
<td>5</td>
<td>Step 4 - Create the Eval Query ....................</td>
</tr>
<tr>
<td>6</td>
<td>Step 5 - Create the Eval Global Query ..........</td>
</tr>
<tr>
<td>7</td>
<td>Step 5 - Link the FORM2626 to the RBVU_DISTINCT_KTR_DUNS4 query</td>
</tr>
<tr>
<td>8</td>
<td>Step 6 - Design the Report .............................</td>
</tr>
<tr>
<td>9</td>
<td>Step 7 - Refer to the finished example for RAP details</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Part XVIII</th>
<th>Example 18 - RMS Construction Calendar</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Overview .............................................</td>
</tr>
<tr>
<td>2</td>
<td>Step 1 - Add a view to create daily records ....</td>
</tr>
<tr>
<td>3</td>
<td>Step 2 - Configure the Queries ..................</td>
</tr>
</tbody>
</table>
Part XIX Example 19 - Mail Merge

1. Overview ................................................................................................................................... 269
2. Step 1 - Configure a Contract Query .......................................................................................... 269
3. Step 2 - Configure a Feature Query ............................................................................................ 272
4. Step 3 - Configure the Design with Richtext and Subreport controls ........................................ 276
5. Step 4 - Configure the Richtext controls for mailmerge, and stretch and setup the shift relative to
6. Step 5 - Add Some Mail Merge Text .......................................................................................... 277
7. Step 6 - Configure the subreport ................................................................................................. 278
8. Step 7 - Add some Richtext for the summary section ................................................................. 279
9. Summary .................................................................................................................................. 280
RMS Report Writing

Example Report
For Training Purposes

Example
1 Example 1 - RMS Program Type

1.1 Overview

This report is designed to be very simple. It is an introductory report to show how easy it is to build a simple single table report in Report Builder. It shows how to use some of the basic report designer tools. If you’ve never used the RMS report designer before, this is the place to start.

Skills shown:
- Report Builder basics
- Single table query
- How to use the designer

1.2 Step 1 - Add the Report

From RMS select File > Utilities > Custom Report Designer - RMS

You should see the RMS Report explorer screen similar to what is shown below:
A few points to be made about the report explorer. By default, you will be working in your own folder indicated by your login ID. Nobody else (other than systems people) has access to your folder so you can design reports and keep them for your own use. If you have a report that you want to publish to everyone, either, move it, or copy it into the root folder "All Folders" and it will show up on the report menu in RMS. Once its in the "ALL Folders" folder, RMS users can easily run the report without knowing anything about the report designer itself.

To add a report, right click on the right hand window pane of the report explorer and select New Report. The report designer will load with an empty report - ready for you to start your design.
Before you do anything else, select File > Save As and give your report an appropriate name - In this case, use "RMS Program Types"

1.3 Step 2 - Create the Query

The hardest part of creating most reports will be getting the data and structuring it just right for the report you want to create. Every report will have at least one query, but may have many queries that can be linked together in master-detail fashion. Each query may itself, be a join of many underlying tables in the RMS database. For this report we will be keeping it very simple and we will create one, single table query to get the data we want.

All of your query work will be done on the data tab in the designer. From the report designer, switch to the "Data" tab
Now, select File > New and you will see that you have two options:

The query wizard is a simple way to create a query, but it is limited in its options. The query designer takes you straight into the screens that will allow you to build your query. For all remaining examples, we will use the query designer, but for this simple report, select the Query Wizard and click Ok. You will see a list of all available tables in the RMS database. Scroll down and select PROGRAM_TYPE, then click on the right facing arrow to move the PROGRAM_TYPE table into the Selected Tables list, then click Next.
For this report, we want all of the fields from the table so select All Fields and click Next.
We don't need calculations, grouping or special ordering for now, so just hit next until you get to the last screen of the wizard, then click Finish.
1.4 Step 3 - Design the Report Header

Switch back to the Design Tab. Notice that the report is divided up into three sections

Header
Detail
Footer
These are called bands. The report builder tool is a "Banded" report writer and it uses bands to control the design of the report. In this step, we will be working on the title of our report so we will use the Header band.

Let's try to make our header look like a standard RMS report with the USACE castle and report title. To insert the castle, select the image button from the toolbar.

Now, click in the header band and an image item will be inserted. Right click on the image and select Picture from the menu and then find your favorite version of the USACE castle to insert.

(Get a castle here)
Size and Position your image

Next, we will add a title and a line underneath it. Select the Text button (The letter A) from the toolbar and then click in the header band. Click in the text box and change the text to say RMS Program Types.
Click on the line button and then click on the header band and a line object will be inserted - size it and position it in the header band.

Finally, lets insert the date as part of the header. To do that, click on the System Variables tool and then click in the header band. The System Variables allow you to get to things like the date, time, document name and so on. It defaults to the date so just size and position your object on the right side of the header band.
We could make the header a little more elaborate, but we will leave it as is because in the next example, we will learn how to use some pre built templates that already have the header bands built for both portrait and landscape reports. At this point, the header band is complete. You can click on the Preview tab and look at your report.

1.5 Step 4 - Design the Detail Band

Now we'll connect our query to our detail report and start printing some actual data.

Put three DBText objects in the detail band and set the table name and fieldname to the three available fields from our query. Also, position the DBText objects, size them, and set the font as you like. Resize the detail band by dragging it upward to make it smaller.

Save the report and then preview it and you should see all of the available program types from the RMS database.
1.6 Summary

This report is not very polished and we did it the hard way by making the header from scratch. However, it is a good example of how to get started with the report designer and how easy it is to build a simple report. Now that you have built the report, you can run it directly from the explorer without having to go into the designer. To change the design, right click on the report in the explorer and select design, to run it without the designer, double click on it in the designer.
2 Example 2 - Standard Templates

2.1 Standard Templates Provided

We have provided several standard template files to make building reports easier and more uniform for you. Use these templates whenever possible to save time and give your reports a standard look.

You can load these up in an empty report and see what they look like. This is what the standard Portrait template looks like:

Basic Templates

RMSPortraitTemplate.rtm
Generic portrait report with no dataviews loaded

RMSLandscapeTemplate.rtm
Generic landscape report with no dataviews loaded

Templates with a Contract DataView

RMSPortraitTemplateWithContractQuery.rtm
Portrait template with a contract query selecting basic fields and sorting by FMT_CONTRACT_DELIVERY

**RMSLandscapeTemplateWithContractQuery.rtm**

Landscape template with a contract query selecting basic fields and sorting by FMT_CONTRACT_DELIVERY
Example Report
For Training Purposes
3 Example 3 - RMS Program Types (redone)

3.1 Overview

In this report we will create a list of program types from RMS just as in example 1, but we will use a pre built template to make things easier and more standardized. Also, we will add a few minor embellishments like interpreting a field value of Y or N to display Yes or No and add some dynamic coloring as well. We will also sort our data in the query designer. Finally, we will use this report to demonstrate many of the shortcuts that the designer has to make sizing, positioning and aligning controls easier and more uniform.

Skills shown:
- How to Load a Template from a file
- Using the query designer
- Using the designer to help with object positioning, alignment and sizing

3.2 Step 1 - Load up the Standard Portrait Template

In example 1, we started with an empty report and created a header on our own, This time, we'll get smart and load our basic report format from an existing file that contains all of the essentials for building a new report.

Create a new report and save it using Save As - Give it a name of “RMS Program Type Listing”

Once the designer comes up, make sure you are on the design tab and then select File > Load From File and then find the RMSPortraitTemplate.rtm file. When the file loads your designer should have an empty report with a standard header and footer.
This is much smarter than starting from scratch every time and it helps to give the reports a common look. The only change we need to make this time is to change the report title. Click on the text object for the title and change it to "RMS Program Types". Save the report.

3.3 Step 2 - Create the Query

Click on the Data tab and then select File > New from the drop down menu. Select Query Designer and click Ok.

The query designer will be displayed and will show a list of all available tables in the RMS database. Scroll down and select the PROGRAM_TYPE table and move it into the selected table list.
Click on the Fields tab and move all three of the available fields into the Selected Fields list.
We won't do any calcs, grouping or searching, but let's sort the data by the code used for each program type. Click on the Sort tab and move the PROG_TYPE field to the Selected Fields list.
Now we have our query with all of the data we need and it will be sorted by code. Click Ok to close the query designer.

### 3.4 Step 3 - Add DBText controls to the Detail Band

Switch back to the detail band and put three DBText controls in the detail band. Connect each one to the appropriate fields using the Table and Field drop downs.

**DON’T SIZE AND POSITION THE CONTROLS YET - WE WILL SHOW SOME SHORTCUTS ON HOW TO MAKE IT EASIER**
3.5 Step 4 - Multi-Selecting and changing properties

The DBText objects that we inserted for our data probably defaulted to the font that we used for the report title. Obviously we want a smaller font. Rather than selecting each object and changing its font, you can select all three and then change the fonts all in one step.

To start off with nothing selected, click on the background of one of the bands so that nothing is selected.

Ways to multi-select:

1) Hold down the Shift key and click each control one at a time.
2) Lasso all three of the controls by left clicking with the mouse and then dragging to form an enclosing lasso around the three DBText items.

Once all three of the objects are selected, change the font to Arial 8 and get rid of the bolding. You will see the actions work on all of the items selected.

While you have the items still selected as a group, move them around by clicking on one of them and dragging them - they will all move as a group.
3.6 Step 5 - Micro moving and Micro Sizing using the Keyboard and Mouse

You can fine tune the position of the controls by using the ctrl key and the arrows on the keyboard. To do that, make sure you have the appropriate controls selected and then hold down your control key and move them around with the arrows on the keyboard.

Note that if you hold down the shift key instead of the control key, then the arrows will change the SIZE of the selected objects - You can increase or decreases the height and width of controls this way.
You can also use the Nudge buttons to control the micro positioning of the controls. With some objects selected, just click on the appropriate Nudge button on the toolbar.

The nudge controls are on the toolbar:

![Nudge Buttons]

3.7 Step 6 - Aligning Objects

When you put objects inside of a band, you usually want to align them in some way. For our example, we want our DBText objects to be aligned so that the print on the same row across the page. You can align them manually using the mouse or keyboard, but the better way is to use the alignment buttons on the toolbar.

![Alignment Buttons]

These allow you to quickly and accurately align items. With your 3 DBText items selected, click on the Align Bottom button and your objects will be perfectly aligned. Some other interesting options are center in band and evenly space. Experiment with those options.
3.8 Step 7 - Size and Position the DBText Items

Use the designer features to size and position the DBText objects as shown below:

Save your report and preview it to see how the data looks.

3.9 Step 8 - Add Column Titles

On the design tab, click the Text button and add a new text object in the header section. Set the text to "Program Type", right click on the text object, and turn on Word Wrap. Set the font to Bold and select the Centered Text option from the toolbar.
Add column headers for the other two columns. You can use Copy (Ctrl+C) and Paste (Ctrl-V) to make things easier.

Save the report and preview it.

### 3.10 Step 9 - Embellishments for the Y/N column - RAP code

The report is pretty good, but let's add some polish. Instead of just printing Y or N, let's make it print Yes or No and while we are at it, let's color the No records red so they stand out more. Report Builder makes this possible by allowing us to write RAP code for events as they happen when the report is running. This is called event driven programming. All we do is find the appropriate event to use and then write our rap code.

Click on the Calc tab of the designer. There you will see a tree diagram of your report and its objects.
On the right hand side is a list of events for the selected object.

Highlight the DBText3 item in the tree on the left. That's the one for our Y/N column. Next, right click on the OnGetText event in the Events pane and select New to create an OnGetText event. You should see a new programming block created where we can now write our event handler.
What is happening here is that every time DBText is rendered in the report, this event is fired giving us a chance to set the text and manipulate the object. To set the text and handle the color, insert this code into the on get text event.

```
begin
  if PROGRAM_TYPE['SHOW_RECORD'] = 'Y' then
    begin
      DBText3.Font.Color := clBlack;
      Text := 'Yes';
    end
  else
    begin
      DBText3.Font.Color := clRed;
      Text := 'No';
    end;
end;
```

Then, right click on the editing window and select Compile. If you did it perfectly, you should see 0 errors.
Now, preview your report and you should see the Yes No in place of the Y/N and the No records should all be Red.

3.11 Summary

This is a much more polished version of the Program Types report. It was also simpler and it showed how to use some of the more advanced features of the report writer in simple ways to make a professional looking report.
Example 4 - Interest Payments

4.1 Overview

This report is pretty straightforward, but does illustrate how to use 2 pass and lookahead for showing totals in headers before the details are printed. It prints out the interest payments for all contracts and totals the interest at the contract level and for the entire report.

Skills shown:
- Sub totalling and Grand totalling
- 2 pass logic and lookahead
- Group Headers
- Adding a Summary Band

4.2 Step 1 - Load the Landscape Template and Set the Basic Design

Load the standard landscape template. Change the title and add column labels as shown.

4.3 Step 2 - Create the Query

On the data tab, select File > New > Query Designer and click Ok.

Select the CONTRACT table and then the RECEIPT_VOUCHER table from the list. When you select the RECEIPT_VOUCHER table, you will see the join dialog. Set the options as shown here:
On the fields tab select the following fields.
On the search tab, set the following filter:
And finally, on the sort tab, set the sort as shown:
### 4.4 Step 3 - Design the Detail Band

Switch to the design tab and add DBText objects and position them as shown here:

![Image of report design](image1.png)

### 4.5 Step 4 - Add a Group

Click on Report > Groups form the pull down menu.

Add a group for the contract_delivery with settings as shown here:

![Image of report groups](image2.png)
Drag down the group header band to make some room and format the group header as shown here:

The INTEREST_AMT object is a DBCalc. Once you have it positioned, right click on it and select calculations and set it as follows:
4.6 Step 5 - Add a Summary Band

Select Report>Summary. A new summary band will open. Set the layout as shown below using a label and another DBCalc. The DBCalc should be set to use the INTEREST_AMT field and to never reset.

4.7 Step 6 - Set 2 pass and lookahead options

If you run the report at is, you will probably see that the total in the header band is not correct. That's because the header gets rendered before the detail data gets printed. At that time, it does not know the total for the entire group. To fix this, we will use 2 pass and lookahead.

Select Report > Pass Setting and make sure Two Pass is checked. This will cause the report engine to make two passes over the data so that our header will know what the total of the detail band is when it gets rendered on the second pass.
Now, we have to tell the report engine that we want our group header total to have the total by setting it to use lookahead. Right click on the INTEREST_AMT dbcalc in the header band and make sure Lookahead is selected.
Now, the total in the header will be the total for the detail rows within its section.

4.8 Summary

This simple report showed how to use group headers and how to enable 2 pass and lookahead to include totals in the header band.
RMS Report Writing

Example Report
For Training Purposes

Example

V
5 Example 5 - AutoSearch

5.1 Overview

The query designer will let you specify all kinds of search/filter conditions. But what if you want to let your user decide what conditions to use. That's where autosearch comes in. Report builder lets you tag search conditions as autosearch and then it will prompt the user for values when the report runs.

This simple listing of contracts is just designed to show how to use some of the different Autosearch features and how to display the selected options in the actual report.

Note the selection criteria displayed in the Summary section above.

Skills shown:
- Autosearch Basics
- Mandatory Autosearch
- Getting a description of the Autosearch selections

5.2 Step 1 - Load the Standard Portrait Template

Switch to the Design tab and load up the standard Portrait Template
5.3 Step 2 - Set the Report Title and Column Headings

Change the report title to Contract List and add three Labels for column headings as shown here:
5.4  Step 3 - Create the Query

Our query will be extremely simple. We will just use the contract table and a few selected fields and we will sort it by FMT_CONTRACT_DELIVERY as usual. The interesting part of this example will be the Search conditions.

Switch to the data tab and select File > New > Query Designer and click Ok

Select the contract table from the list
Switch to the fields tab and select the fields as shown.
Finally, switch to the sort tab and select the FMT_CONTRACT_DELIVERY field to sort by

\[ 5.5 \]

**Step 4 - Add a Between Autosearch**

Autosearch lets you give the user control over what records are included in the report when it prints. There are several different types of autosearches including between, in list, less than, greater than, blank and so on.

Let's start by adding a between condition on the award date so the user can filter down to just contracts awarded for a certain period of time.

Switch to the Search tab and select the AWD_ACT field from the list. Set the operator to Between and leave the value blank. Set autosearch to Yes.
When the report runs, the user will be prompted to enter a from / thru date for the award.

### 5.6 Step 5 - Add an In List Autosearch

An In List search allows the user to add as many entries as they choose to a list. If the record matches any item in the list, then it gets included in the report.

Let's add an In List on Program Type:

Select the PROGRAM_TYPE field from the list and set the options as shown here. Also, check the Mandatory block. That will make it so that the user has to make an entry before the report runs.
5.7 **Step 6 - Add a Like Autosearch**

A like autosearch does partial pattern matching. It lets the user search for matches where the beginning of the field matches some value. We will do a Like on the contract number so the user can type in DACA83 or something of that sort to match certain contracts.

Select the CONTRACT_DELIVERY field from the list and set the options as shown here
5.8 Step 7 - Add a Greater Than Autosearch

Finally, let's let the user filter to only contracts greater than a certain amount. Select the CURRENT_CONTRACT field from the list and enter the options as shown below.

Note that 100000 is entered as the value. That will become a default value when the user runs the report.
5.9 Step 8 - Build the Detail Band

Let's throw some fields into the detail band so we can see how our autosearch is working for us. Drag a region from the toolbar and set it to parentheight/parentwidth and then put 4 DBText objects into the region setting up the fields and positions as shown here.
5.10 Step 9 - Test The AutoSearch

Save the report and close down the designer. Then, run the report by double clicking on it in the explorer and you should see this screen. Note the default value kicked in and the PROGRAM_TYPE entry is mandatory.

Enter some values to see what results you get on your report.
5.11 Step 10 - Add the Search Text to the Report

We have one glaring problem remaining. When you run the report, nothing prints on the actual report showing the user what the conditions are. Let’s fix that with some RAP code that fills a memo with the search text.

Return to the designer. From the Design tab, select Report and then select Summary to add a summary section band. Add a regular Memo to the summary band as shown. Right click on the memo and make sure Stretch is selected so that the memo will grow as needed.
Now, let's add the RAP code. Click on the Calc tab. Highlight the report object and add a BeforePrint event with this code

```csharp
Report.GetAutoSearchDescriptionLines(Memo1.Lines);
```
This will fill the memo with a description of all of the options selected in autosearch.

5.12 Summary

This report showed some important steps in allowing the user to set filter criteria and in showing that criteria on a report.
6 Example 6 - Special AutoSearches (Contract and Office)

6.1 Overview

In the previous example, we showed how autosearch works. In this example, we show 2 special cases for the autosearch. One for contracts and one for offices.

Because contracts and offices are so integral to RMS, we have built in special autosearches for them allowing the user to pick from a list.

Skills shown:
- Special Autosearches for Contract and Office

### Step 1 - Load the existing Report

We are just demonstrating some additional search features so we will use an existing report. Load the report from Example 5.
6.3 Step 2 - Configure Contract AutoSearch

Switch to the data tab in the designer and open the query designer.

Switch to the search tab and remove all of the existing searches.
To configure the special contract autosearch, select contract_id from the list, set the operator to In List and set AutoSearch to yes. These setting will cause the special contract selection screen to appear when the user runs the report.
6.4 Step 3 - Test the Contract AutoSearch

Save the report and close the designer. Run the report from the explorer and you will see the special option allowing the user to pick from a list of contracts.
6.5 Step 4 - Configure Office AutoSearch

Go back to the designer and as with the contract search we did before, let's configure office autosearch.

Clear out the existing search and set up the search as follows:
Example 6 - Special AutoSearchs (Contract and Office)

This will cause the special office lookup to be available to the user.

Before closing the query designer, switch back to the Fields tab and add OFFICE_SYMBOL to the list of selected fields:
6.6 **Step 5 - Test the Office AutoSearch**

Switch to the design tab and add the office_symbol to the detail band as shown here so that we can properly test the office lookup.
SPECIAL NOTE ON OFFICE_SYMBOL FIELD: The OFFICE_SYMBOL field is NOT the correct field to use in an actual report. It is an internal code used to link to the offices table. To get the correct field for a report, you have to do a join to the OFFICES table and use the OFFICES.DISPLAY_SYMBOL field instead. (See the example report on doing this simple join)

Save the report and exit the designer. Run it from the explorer and play around with the office lookup:
### Select Contract

<table>
<thead>
<tr>
<th>Office Symbol</th>
<th>Office Name</th>
<th>Office Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>AE SUPPORT</td>
<td>AE Support for Submitted Rev.</td>
<td>PROJECT</td>
</tr>
<tr>
<td>CEPHO-EC-DA</td>
<td>Architecture/Specs Section</td>
<td>PROJECT</td>
</tr>
<tr>
<td>CEPHO-FF-C</td>
<td>Civil &amp; Public Works Branch</td>
<td>District</td>
</tr>
<tr>
<td>CEPHO-EC-T</td>
<td>Civil Works Technical Branch</td>
<td>PROJECT</td>
</tr>
<tr>
<td>CEPHO-EC-DC</td>
<td>Civ/Struct/Geotech Section</td>
<td>PROJECT</td>
</tr>
<tr>
<td>CEPHO-EC-C</td>
<td>Construction Branch</td>
<td>District</td>
</tr>
<tr>
<td>CEPHO-EC-CC</td>
<td>Construction Services Section</td>
<td>District</td>
</tr>
<tr>
<td>CEPHO-CT-C</td>
<td>Construction/A/E Contracting</td>
<td>PROJECT</td>
</tr>
<tr>
<td>CEPHO-CT</td>
<td>Contracting Division</td>
<td>AREA</td>
</tr>
<tr>
<td>CEPHO-EC-S</td>
<td>Cost Engineering Branch</td>
<td>AREA</td>
</tr>
</tbody>
</table>

### Selected Offices

<table>
<thead>
<tr>
<th>Office Symbol</th>
<th>Office Name</th>
<th>Office Type</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Office Record Count: 34  You have selected 0 office(s).

[Image of the Select Contract window with tables and buttons]
7 Example 7 - Office Symbol Join

7.1 Overview

In an earlier example we used CONTRACT.OFFICE_SYMBOL in our report but mentioned that it was a mistake to do so. This short example shows why.

Skills shown:
- Basic Join to get the correct Office Symbol for a Contract

7.2 Step 1 - Load the existing report

Load the report from the example 5
7.3 Step 2 - Join to the Office Table

The reason that we don't want to use the CONTRACT.OFFICE_SYMBOL field is that it is just an identifier used to link to the OFFICES table. It cannot be relied on to be an accurate actual office symbol which can be used in reports. The reason for this is related to the recent change that allows RMS users to change the office symbol for an office. While it may look like you can use this field, trust me, it will be wrong in some cases. For any offices that are added after we made this change to RMS, the CONTRACT.OFFICE_SYMBOL will just be a meaningless number.

Here is a list from SAS - Notice the K6 id's at the end of the list. If you use CONTRACT.OFFICE_SYMBOL, that is what you are going to get.
To fix this, you need to do a simple (but irritating) join to the office table anytime you want the office symbol for a report. ALWAYS use OFFICES.DISPLAY_SYMBOL to display in the report and use CONTRACT.OFFICE_SYMBOL = OFFICES.FLSYMBOL as the join condition.

To illustrate, switch back to the data tab and open the query designer. Add OFFICES to the list of tables along with CONTRACT. When the join dialog poops up, you will have to manually set the join condition as shown below.
7.4 **Step 3 - Add the DISPLAY_SYMBOL field**

On the fields list, add the DISPLAY_SYMBOL field
Then, in the designer, add a DBText and set it to DISPLAY_SYMBOL
7.5 Summary

This quick example shows how to properly join the contract table to the offices table to get the DISPLAY_SYMBOL field for showing in reports.
8 Example 8 - Equipment List

8.1 Overview

This report will list out the equipment for a selected contract. This will be a landscape report and will involve columns with more than one data item. It will also show how to handle a memo text item where the amount of text entered is unlimited. Also, getting the data for this report will involve querying multiple tables and using some SQL expressions to get some of the values needed for the report.

Skills shown:
- Building Landscape Reports
- Using SQL expressions (Case, NVL, and Decode)
- Correlated Subqueries
- Formatting Fields
- Stretching Memos
- Using Regions
- Using Search Criteria to Contract(s)
- Grouping Data By Contract
- Simple Table Join
- Using a Shape as a Group Header Background
- Send to Front/Send to Back

8.2 Step 1- Planning the queries and Finding the Data

Your hardest task at times will be finding the data that you want in the RMS database. For this report, we want data on equipment and its operated hours. In this case, the table name where the data is located is rather deceiving. The table name is actually SAFE_CK. There are a few ways you can try to find the data you are looking for. First, you should print out the RMS data dictionary. You can find it in RMS under Reports>Library>Data Dictionary. Another thing you can try is to go into RMS to the screens where the data you want is located. Select one of the fields and then Ctrl Right Click on it.

Here is the screen that comes up for the hours run edit box on the equipment screen. From this, you can see the table and field name
All of the data we want for this report is available in the SAFE_CK table with one exception. We want to print the first day on site and that is not stored with the equipment item. To get it, we will have to look at the table that stores the actual hours for the equipment - EQUIP_HR. We want to join the SAFE_CK table with the EQUIP_HR table and get the first day on site for each piece of equipment. We will do this by using a correlated subquery in an expression in the query designer.

Also, we will want this report to handle multiple contracts so we will be doing a join in the query designer with the contract table to get the contract number and title.

### 8.3 Step 2 - Load the Landscape Template

Add a new report and save it with the name of Equipment List. Form the design tab of the report, select File>Load From File.
8.4 Step 3 - Create the Query

Switch to the data tab in the report designer and select File > New > Query Designer and hit Ok.

From the list of tables, scroll down to SAFE_CK and move it into the list of selected tables.

Now, switch to the Fields tab and select the fields as shown. These are the fields that we are interested for our report.
In addition to the safe_ck table, we want to get some information from the contract table. Specifically, we want the contract number and delivery number along with the title. To get that data, we need to join SAFE_CK with CONTRACT. Switch back to the tables tab and add CONTRACT to the list of selected tables. A table join dialog box will appear.

**Join Table Dialog:** It's important to understand how this dialog works. Join Type tells the database what to do when it doesn't find matching rows between the two tables. Let's use an RMS example to illustrate. Let's say you want to join the CONTRACT table with the OFFICES table. We know that an Office can have many contracts and that a contract can be assigned to no office, or to one office. Let's say that for our join, that OFFICE is the Left table and CONTRACT is the right table.

**Inner:** If a match is not made between both tables, then no row is inserted in the result set. In our example, if we have an office with no contracts, then the office will not be included. Likewise, if we have a contract with no office, it will not be included either. Only those records where an Office and a Contract match will be included in the result set.

**Examples:**

```
SELECT OFFICES.FLNAME, CONTRACT.FMT_CONTRACT_DELIVERY FROM OFFICES INNER JOIN CONTRACT ON OFFICES.FLSYMBOL=CONTRACT.OFFICE_SYMBOL
SELECT OFFICES.FLNAME, CONTRACT.FMT_CONTRACT_DELIVERY FROM OFFICES, CONTRACT WHERE OFFICES.FLSYMBOL=CONTRACT.OFFICE_SYMBOL
```

(These are equivalent, just using different syntax)

Outer joins do not require there to be a matching row in both tables. There are three types of outer joins:

**Left Outer:** All records from the Left table will always be included - even if there is no matching record in the right table. In our example, all offices will be included even if they have no contracts. If a contract is not assigned to a valid office, however, it will not be included.

```
SELECT OFFICES.FLNAME, CONTRACT.FMT_CONTRACT_DELIVERY FROM OFFICES LEFT OUTER JOIN CONTRACT ON OFFICES.FLSYMBOL=CONTRACT.OFFICE_SYMBOL
SELECT OFFICES.FLNAME, CONTRACT.FMT_CONTRACT_DELIVERY FROM OFFICES, CONTRACT WHERE
```

(These are equivalent, just using different syntax)
Right Outer: All records from the Right table will always be included - even if there is no matching record in the left table. In this case, all contracts will be included even if they do not have a valid office. If an office has no contracts however, it will not be included.

SELECT OFFICES.FLNAME, CONTRACT.FMT_CONTRACT_DELIVERY FROM OFFICES RIGHT OUTER JOIN CONTRACT ON OFFICES.FLSYMBOL=CONTRACT.OFFICE_SYMBOL

(These are equivalent. Oracle allows the use of (+) to indicate which table can have missing records. Think of it like oracle is adding (+) some blank records to get a match)

Full Outer: All records from the Right table will always be included AND all records form the left table will be included - even if there is no matching record in the corresponding table. So, if an office has no contracts, it will still be included and if a contract has an invalid office, it will still be included.

SELECT OFFICES.FLNAME, CONTRACT.FMT_CONTRACT_DELIVERY FROM OFFICES FULL OUTER JOIN CONTRACT ON OFFICES.FLSYMBOL=CONTRACT.OFFICE_SYMBOL

(Oracle's special (+) syntax does not support full outer joins - you must use the explicit syntax)

NOTE: Full outer joins are not functional in the report tool - They work as a full inner join due to a bug with the report designer.

To make this dialog clear as far as knowing which type of join to use, always remember that the report writer is treating the table in the combo box (SAFE_CK) in this case as the LEFT table and that the table on top of the dialog box (CONTRACT in this case) is the RIGHT table.

If you are still confused about joins or want more examples, you may want to check out Wikipedia link on SQL Joins, Easy Oracle SQL Book

So back to our actual join for this report, we only want to join based on the contract_id field and we want to use an inner join. Make your join dialog look like the one below and click Ok.
Now that our tables are joined, jump over to the fields tab and select the FMT_CONTRACT_DELIVERY field and the PROJ_NAME field form the contract table as shown here:

Now that our tables are joined, jump over to the fields tab and select the FMT_CONTRACT_DELIVERY field and the PROJ_NAME field form the contract table as shown here:
8.5 Step 4 - More Query Work - SQL Expressions

In this step, we will refine our query in 3 ways:

- We will show another way to handle the Y/N fields so that they display Yes or No.

- The report shows the last inspection. What we want to do is use the RE_INSPECT field if it has a value, otherwise, we will use the INSPECTION field.

- We are still missing the first day on site, which is not stored with the equipment. We will have to use an oracle expression to get that from the table that has the hours in it.

We'll start with the Y/N field. As before, we don't want our report to show Y or N, we want Yes or No instead. In the query designer, switch over to the Calcs tab and move the COND_ANS field into the selected list, select Expression from the function type dropdown, and past in this oracle expression:

```
decode (cond_ans, 'Y', 'Yes', 'No')
```

This is an Oracle function that takes its first parameter and then starts comparing it to pairs of parameters. When the first parameter matches the first parameter of a pair, the result will be the second parameter of the pair that matched. In this case, the decode is saying if cond_ans has a Y, then return Yes else return No.

Click on the field alias column and change the name of our calculated field to V_COND_ANS.

Now, lets handle the last inspection date. This time we will use Oracle's NVL function which allows you to specify a value to return in case the first parameter is NULL. Select the RE_INSPECT field and
move it into the selected list, set function to Expression, and past in this code

NVL(safe_ck.RE_INSPECT,safe_ck.INSPECTION)

Then click on the first column and change the alias to V_REINSPECT

Finally, we need to get the first day on site from the equip_hr hours. What we want is the earliest date that hours were reported to get this, select the inspection field and move it into the list of selected fields. Then change the function type to Expression and enter this SQL

(select min(insp_date) from equip_hr where contract_id=safe_ck.contract_id and item=safe_ck.item)

This expression will become a correlated subquery and will get the earliest date for each row in the main query. Change the field alias to V_FIRSTDAY.
8.6  Step 5 - Enable the query to allow the user to select contract(s)

In this step we will set up the query to let the user pick a contract when the report runs. Or, they can pick multiple contracts. This is a simple task. Switch over to the Search tab on the query designer and move the SAFE_CK.CONTRACT_ID field into the selected list of fields. Then, set the operator to In List and set AutoSearch and Mandatory to Yes as shown here.
This is a special setup that will tell the report engine to popup a dialog and ask the user which contracts to include in the report when it gets run.
8.7 Step 6 - Sort the query

The final step for the query is to get it into the proper sort order. Since the user may select multiple contracts, we will be adding groups to the report to separate the equipment by contract. Within a contract we want the items sorted by equipment ID number.

Switch to the Sort tab and select CONTRACT_ID and then select ITEM.
8.8 **Step 7 - Add Column Titles**

Save the report and then switch over to the Design tab.

Add labels to the blue column header for each of the data columns that we will use. Add lines as shown below for the hours section of the report.
8.9 Step 8 - Add a Region for the detail Band

Anytime you are going to have a memo or other variable height object, ALWAYS use a region within the band. This will eliminate some quirky behavior when your detail band is near a page break. By putting all of your detail items in a region, you will have more control over keeping the items grouped together.

To add a region, click on the Region toolbar button and drop a region into the detail band. Once you have the region in the band, right click and select ParentWidth, ParentHeight, Stretch and KeepTogether. This will make the region occupy the entire detail band it will set it up to stretch with our notes memo.
8.10 Step 9 - Add the DBText Objects

Now, add the DBText objects to print the data within the region on the detail band. Add all of the fields as show below - Do not add a DBText object for the notes, it is special. Adding the DBText objects is straightforward. Just drop a new DBText object into the region and then set the field to the appropriate field from the query for each item.

8.11 Step 10 - Add the DBMemo Object

Now, add a DBMemo object for the notes field and position it as shown.
In order for the report engine to allow the DBMemo to grow vertically as needed, you have to set it to stretch. Right click on the DBMemo and select stretch. Now, when the report runs, the dbmemo object will size vertically as needed based on the text in the database.

### 8.12 Step 11 - Add Grouping by Contract

To clearly separate the data by contract, create a group in the report. A group allows you to create a header and footer to enclose sections of the report. In our case, we want to have a header every time we start a new contract.

From the Report pulldown menu, select Groups

![Image of Report Designer](image)

From the groups dialog box, select the FMT_CONTRACT_DELIVERY field and set the options as follows:
Note: Because groups are based on a change in field value, the field on which a group is based should always be used to order the records. You may remember that the query for this report is ordered by the ‘fmt_contract_delivery’ field. You must always order the records by the fields you intend to use for groups because ReportBuilder will not sort records for you.

The options on the group dialog box are self explanatory. If you want each group to start on a new page, you can check the box “Start new Page”. Keep in mind, that this may lead to a lot of white space on your reports and will make your reports take up more pages. Keep together is an option that tries to keep the data for a group together as much as possible. Again, this may lead to more white space on the report as the engine might decide to start a group on a new page to keep it together.

Click Ok to add the group.

8.13 Step 12 - Design the Group Header

Now that we have added a group, we have to design the header for the group. In our case, we just want to print the contract number and title for each contract.

Notice, that when you add the group, there is no space created for your group header. To add some space, grab the group header with your mouse and drag it downward.
After making space for the header, the group header band should look like this:

Now, follow these steps to design the group header.

- Add a shape object for the background of the group header. Set it to parentheight and parentwidth (right click for these settings) and set its color to grey.
- Add a DBText object and set it to use the FMT_CONTRACT_DELIVERY field. Make the font Bold to make it stand out more.
- Add a DBText object for the contract title and set it to use the PROJ_NAME field

Your design should look something like this:
8.14 Summary

Save your report and preview it. Also, be sure to run it from outside the designer to see how the contract selection screen works.

This report showed more advanced concepts including some oracle expressions and grouping. It also showed how to use a region and how to make a DBMemo stretch with the actual data.
9  Example 9 - Contract Value History

9.1  Overview

This report will show the entire history of a contract's value in chronological order. It will include a running total so you can see the value of a contract at each point in time as well as a grand total for the current contract amount. This is an interesting report in that it needs to get data from award and for mods and treat them like they are one dataset. To do this, we will create a view. Also, we will create our first subreport and this report will have 2 data pipelines joined together in the designer.

Skills shown:
- Multiple datapipelines and linking pipelines together
- Creating a custom view for a report
- SQL Union
- Subreports
- Running Totals
- Grand Totals
9.2 Step 1 - Load the Standard Portrait Template and Set the basic report items

From the designer, load the standard report template into the designer. Change the title of the report and add column headers for our 4 columns of data as shown:

![Image of Report Designer]

9.3 Step 2 - Create a View for the detail data

We want to combine data for award and mods and print that data out in our report. Unfortunately, the award data is in the contract table and the mod data is in the MOD_LOG data. In this case, we don't want to do a simple join where we combine mod data and award data into the same row, instead, we want the award data to have its own row and the mod data have its own rows as well. What we need is an SQL Union. A union takes to like columned result sets and combines them so that they look like one result set. The trick here is to make the contract table data and the mod_log table data look like columned. Here is the SQL to do it.

```
select contract_id, 'Contract Awarded' as v_text, awd_act as v_date, award_contract as v_amt from contract
union
select contract_id, mod_no||' '||mod_ds as v_text, govsig_date as v_date, mod_tot as v_amt from mod_log
```

We are just tricking the database into treating the two tables like they have the same columns and then we are unioning them together. To make that SQL usable in our report, we will create a view. You should use a view whenever your SQL gets too complicated for the report engine to handle. To add a view to our report, switch to the data tab and select File > Create View.
The view browser will be displayed. Click on add view:
Name the view VALUE_HISTORY and paste this SQL into the SQL editor

```sql
select contract_id, 'Contract Awarded' as v_text, awd_act as v_date, award_contract as v_amt from contract
union
select contract_id, mod_no||' '||mod_ds as v_text, govsig_date as v_date, mod_tot as v_amt from mod_log
```
Click on create view and your view will be created. Now, when we return to the query designer, we will be able to use this view just like any other table.

### 9.4 Step 3 - Create 2 Data Pipelines

We will create two data pipelines for this report. One with the basic contract information and the other with the details contract value information. This way we can create a subreport for the detail data and total control over it.

Create a new query using the query designer. Select the CONTRACT table and chose the three fields as show below:
Next, create a second datapipeline using the RBVU_VALUE_HISTORY view we created earlier and select the following fields:

Now, join the two pipelines together by grabbing the CONTRACT_ID field in the RBVU_VALUE_HISTORY table and dragging it over to the CONTRACT_ID field in the CONTRACT table. This will create a link as shown below. This means that each time the contract table traverses a record, then the RBVU_VALUE_HISTORY will traverse all records where the CONTRACT_ID matches the current CONTRACT_ID in the contract table.
9.5  **Step 4 - Add a subreport for detail data**

Next we are going to add a subreport for our detail data. This will allow us to treat the detail data like its is its own separate report so we can get it just the way we want. Think of a subreport as a report within a report.

Switch back to the Design tab and drop a subreport into the detail band. You will notice a new tab at the bottom of the designer for the subreport. You can switch back and forth between the main report and your subreports by clicking on these tabs:
Now, click on the SubReport tab - Everything you do at this point will be in reference to the subreport.

- Click on Report > Data and make sure that the subreport is set to use the VALUE_HISTORY pipeline.
- Make room in the title band, add a background shape and color it and then add DBText objects for the FMT_CONTRACT_DELIVERY and PROJ_NAME fields
- In the detail band, add DBText objects for V_DATE, V_TEXT and V_AMT fields: at this point your designer should look something like this
9.6 **Step 5 - Add a running subtotal**

Now let's add a DBCalc to take care of our running subtotal.

Select a DBCalc object from the toolbar and position it in the right side of the detail band as shown.
Right click on the DBCalc and select Calculations. Make sure Sum is selected and make sure that Reset Group is empty. Remember, this is a subreport so, we never want the total to clear out. It will clear out on its own because the subreport will be a new report for each contract. By not clearing it, we will create a running total. In the Field dropdown in the designer toolbar, make sure the field is set to V_AMT since that is the field we want to keep running a total on.

9.7 **Step 6 - Add a grand total Summary Band to the SubReport**

To finish up the SubReport, add a summary band that has the final total for the contract amount.

- Grab the summary band and drag down to make room.
- Add a Label for the Current Contract Amount title
- Add a DBCalc and set the field to V_AMT, and the Calculation to Sum with no reset
- Add a line above the text in the summary band
9.8 Summary

This report introduced the import ability to use views in a report when the SQL gets too complicated for the report writer to handle. It also showed how to do a basic subreport and work with multiple data pipelines. To enhance the report further, you could add a filter and sort to make the report more useful.
10 Example 10 - Office Org Chart

10.1 Overview

This report lists out the RMS office structure in hierarchical office order. It indents child offices in successive levels from the parent. It’s a rather gimmicky report, but it shows how to manipulate placement of report objects at runtime.

Skills shown:
- Getting creative with Oracle Functions
- Anchoring report objects
- Moving and sizing regions at run time with RAP
- Learning about RMS office fields and structure (OFFICE_SEQUENCE)
10.2 Step 1 - Load the Portrait Template and Set Report Title

Load the standard template report and set the Title to Office Org Chart

10.3 Step 2 - About the Office Hierarchy

The offices in RMS are shown in a hierarchical fashion where an office finds its place in the tree based on who its parent is. That's nice for the end user, but difficult to deal with for programmers. Oracle actually does have a hierarchical SQL methodology, but it's too generic for use to use. There is no magic SQL method to say give me the offices and order by RMS OFFICE ORDER.

Thankfully, the staff at the RMS center have provided a solution via the OFFICES.OFFICE_SEQUENCE column.
The OFFICE_SEQUENCE column is set by triggers and stored procedures in the database in such a way that when you order by OFFICE_SEQUENCE, you will get the offices in the same order as the RMS office tree. The OFFICE_SEQUENCE is a series of numbers indicating where in the tree an office should go - For example 001.001.002 means that this office is the 2nd suboffice of the first suboffice of the root office. The first root office is always 001. If your district has more than one office at the root, they will be 002 and so on.

This makes it easy to order offices and to find all of the suboffices for an office. For example to get all of the suboffices of 001.001.002, just use a OFFICE_SEQUENCE like '001.001.002.%' clause.

### 10.4 Step 3 - Create the query

From the data tab, select File > New > Query Designer and click Ok.

Select the OFFICES table into the selected list.
Switch to the fields tab and select the fields as shown here. We will use all of these in our report.
Switch to the Sort tab and sort the offices by OFFICE_SEQUENCE as we have already discussed.
10.5 Step 4 - Create a Calculated Field for the Office Level

We want to get a little bit fancy and indent our report based on the level that the office falls in the tree. We've discussed the OFFICE_SEQUENCE field and we know that it is based on where it falls in the tree, so we will use that to figure out our level. If we want the indentation level to be zero for the root of the tree, then we can use the length of the OFFICE_SEQUENCE to figure out the level by adding one to the length dividing by 4 and then subtracting one from the result. Here is the SQL expression to do it:

$$\frac{(\text{LENGTH(OFFICE\_SEQUENCE)+1})}{4}-1$$

Switch to the Calcs tab of the query designer and drag down the OFFICE_SEQUENCE field and set the new calculated field as shown here.
10.6 Step 5 - Layout the Detail Band

Add a Region to the detail band and then add labels and DBText objects as shown here
10.7 **Step 6 - Dynamically indent the Region**

To give the rows a bit of an office tree feel, we will indent the region based on the office level.

We can do this by coding a RAP event for our detail band. We will simply set the Left property of the Region based on the tree level and then we will adjust the width of the region so that its right side always ends at the same location. Here is the code to do that where we are moving in a quarter of an inch for each level in the tree,

\[
\text{Region1.Left := (OFFICES['OS_TREE'] * 0.25)};
\]
\[
\text{Region1.Width := 8.0 - Region1.Left};
\]

Switch to the Calc tab, highlight the Detail band, and add a BeforePrint RAP event.

![Image](image-url)  

10.8 **Step 7 - Anchor the DBText Objects**

By default, objects are anchored using the top and left anchor point. When you start playing games like resizing regions at run time, this can lead to problems. All of our report objects that are anchored using Left, are going to shift when we move the left side of our region. We want that for the objects on the left part of our region, but the objects on the right side of our region should stay in the same relative position. To accomplish that, we need to change the Anchoring scheme for those objects.

Switch to the design tab and Select View > ToolBars > ReportTree from the PullDown menu.
With the Report Tree Open, select the DBText Object for the DISPLAY_SYMBOL and scroll down to the Layout > Anchors Section
Change your anchoring scheme from using Left to Using Right instead. This will insure that this object keeps the same relative Right position within the region even when it is resized.
Repeat the Right Anchoring steps for the remaining objects on the right side of the Region.

10.9 **Summary**

This report showed some creative ways to give a static page a more dynamic tree like feel by indenting regions. It discussed anchoring and using the RMS office fields to sort offices in RMS office order.
RMS Report Writing
Example Report
For Training Purposes

Example XI
11 Example 11 - Progress Payment Processing

11.1 Overview

This report is a summary style report that an office can use to see what payments need to be processed for the current month. The report lists all awarded contracts that should have a payment for the current month and shows an entry for every report whether it has a payment or not. We will have to do an outer join to accomplish this. The report makes quite a bit of use of Oracle date functions. Very importantly, this report will illustrate how to properly use an OR condition in the WHERE clause of the query. Buckle up your seat belts because there is quite a bit of RAP code as well.

Skills shown:
- How to properly use ANDs and ORs in the WHERE clause
- Oracle date functions and SYSDATE
- Using an expression in the FIELD portion of a search condition
- Using Variable Objects
- Outer Join
- Renaming objects using the report data tree
- Dynamically setting object visibility
- Joining Multiple tables
- SQL groups
- RAP Code
- Creating Query Links with Multiple Fields
- Multiple Pipelines in One Report

11.2 Step 1 - Load the Standard Landscape Template and Design the Header

Create a new report and load the standard Landscape template. Change the report title and layout the column headers as show here:
11.3 Step 2 - Create a Left Outer Join DataView

We want to have a row for every contract that is awarded and not yet final paid. For each of these contracts, we want to show the payment information for the current month. To do that, we will join the CONTRACT table to the REQUEST table. However, we want to make sure we don't eliminate contracts just because they do not have a payment entered for the current month yet. The outer join is what we need.

From the data tab, select File > New > Query Designer and click Ok

Select the CONTRACT table and put it into the selected tables list. Then, select the request table and put it into the selected list. The Join Table Dialog appears:
First off, we don't want to use its suggested fields to do the join. It suggested them because the field names match in the two tables. We just want to join on the CONTRATC_ID field, so remove all of the others.
If we keep the join as an inner join, then contracts without a matching request row are going to get eliminated and we don't want that. So, switch this to a Left Outer join instead. That will tell the engine to keep ALL of our rows in from the contract table even if there is no match from request.
11.4 Step 3 - Filter the Contract Rows

We only want contracts that are awarded and have not been final paid before the current month. Those are all the contracts that we expect to have some kind of payment for the current month.

Switch over to the search tab. Select the AWD_ACT field and set its requirements to Not Blank. This will eliminate any contracts that have not yet been awarded.
The next condition is more tricky. We want to filter out any contracts that have final pay dates, but not those that are final paid in the current month. We will use the Oracle TO_CHAR function and the built in sysdate variable to get the rows we want. Select the FCOMP_ACT field and put it into the selected criteria list. Set the Operator to Blank. Select the FCOMP_ACT field again and this time, set the operator to > and the value to

\[
\text{TO\_CHAR}(\text{sysdate},'\text{YYYYMM}')
\]

click on the field alias and change it to

\[
\text{TO\_CHAR(\text{CONTRACT.FCOMP\_ACT},'\text{YYYYMM}')}
\]

This says we want rows where the FCOMP_ACT is blank or where the FCOMP_ACT month and year exactly match the month and year of the system date. However, by default, all of the criteria we add will use AND logic and in this case we want OR logic.

Right click on the TO_CHAR(... field and select the Insert OR option from the context menu.
Your screen should now look like this:

<table>
<thead>
<tr>
<th>Field</th>
<th>Operator</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>CONTRACT.AVAIL.ACT</td>
<td>NOT BLANK</td>
<td></td>
</tr>
<tr>
<td>CONTRACT.FCOMP.ACT</td>
<td>BLANK</td>
<td></td>
</tr>
<tr>
<td>OR</td>
<td></td>
<td></td>
</tr>
<tr>
<td>TO_CHAR(CONTRACT.FCOMP.ACT, 'YYYYMMDD')</td>
<td>&gt;</td>
<td>TO_CHAR(update.'YYYYMM')</td>
</tr>
</tbody>
</table>

If we left things as they are however, we would be making a BIG mistake. And one that even experienced SQL folks sometimes make. Right now, our query is basically of this form:

A AND B OR C

In SQL, AND has precedence over OR so with this form, anytime C is true, the expression will be true REGARDLESS of A. What we need is this form:

A AND (B OR C)

In this form, A must be true AND either B OR C must be true. This is an important difference. To get the form we want, we have to insert parentheses at the correct spot. Right click on CONTRACT.FCOMP.ACT and select Insert Parentheses and you should get this:

The begin and end will become parentheses when the report engine sends the query to the database. The begin is in the correct spot, but we need the end to be after the TO_CHAR(...) entry. Use the arrows on the left side of the window to move the end downward in the list until you have this:
11.5  **Step 4 - Filter the Request Table Rows**

We have the correct contract filter, but as the query stands now, we will be getting a joined row for every payment on a contract. We want to filter to get only the last payment.

Select the INV_NO field from the list and add it to the list of selected criteria. Set the operator to =. The tricky part here is that we want the INV_NO to be the biggest INV_NO for a given contract. So what we compare our equality to is this correlated subquery:

\[(SELECT \text{MAX(INV\_NO)} FROM \text{REQUEST} \ WHERE \text{CONTRACT}\_\text{ID}=\text{CONTRACT}\_\text{CONTRACT}\_\text{ID})\]

which returns the highest invoice for a given contract.

Your screen should now look like this:

11.6  **Step 5 - Select the fields and the sort order**

To finish off this query, let's select our required fields and sort order. Switch over to the Fields tab and select the fields as shown here:
Finally, switch to the Sort tab and sort the query by FMT_CONTRACT_DELIVERY

11.7 Step 6 - Create a second query for the Receipt table

Our contract/invoice query is set now, but we still need some more data. We have to use the RECEIPT_VOUCHER table to get payment amount, check date and interest amount information.
Save your CONTRACT dataview and then select File > New > Query Designer and click Ok.

From the list of tables, select RECEIPT_VOUCHER and then switch to the Fields tab and select the CONTRACT_ID and INV_NO fields as shown. These are the only two fields we need because the rest will be handled on the Calcs tab. We need these two, because we will be linking to the CONTRACT dataview using them.

11.8 Step 7 - Create Calc Fields for the Request Pipeline

The RECEIPT_VOUCHER table may have many rows for a single payment. We don't want that level of detail. We just want a payment date, a payment amount and an interest amount. To get those, we still use Oracle group functions and we will group our table by invoice number.

Switch to the Calcs tab and enter the three group calcs as shown here:
Now instead of having three or four receipts for an invoice, these group expressions will just give us the sums and max values as we have requested.

11.9 Step 8 - Check the Grouping of the Query

If you use group expressions, then your query cannot return detailed information for the fields involved in your group calcs. In other words, we can’t say we want the SUM(NET_AMT) and then also say that we want NET_AMT itself. So, we have to make sure we group appropriately. Fortunately, the report tool will generally take care of this for you and automatically create your groups.

Switch to the Groups tab and you should see this:
This is saying to group by CONTRACT_ID and INV_NO so our sums and max values will now be for a given contract/invoice which is exactly what we need.

**11.10 Step 9 - Linking the DataViews**

At this point we have 2 dataviews. One for the Contract/Invoice and another for the Receipts for an invoice. Now, we need to link them together.
Grab the CONTRACT_ID field with your mouse in the RECEIPT_VOUCHER dataview and drag it over to the CONTRACT_ID in the CONTRACT view. That makes the link between our two dataviews. But, we need to link by CONTRACT_ID AND INV_NO. If you try to create a second link by dragging the INV_NO field, you will see that your first link vanishes. Since we need both, we have to edit the link. With your mouse, double click on the link created:

When you double click the link, you will get the link editor. Set the options as shown including the #2 radio button which makes the link work like an outer join.
11.11 Step 10 - Add a region and DBText Field to the detail band

Switch to the design tab and drop a region into the detail band. Set it to ParentHeight and ParentWidth. Some of our detail band will have to be done using variables and calcs so for now, just add the following DBText objects:
Set the fields of each of these DBText objects to the appropriate field - They are self explanatory.

11.12 Step 11 - Add a Variable Object for the Invoice Number

We have a special requirement for our invoice number. If there is no invoice for the current month, then we want to show the next available invoice number and add a red label indicating that the invoice is projected. IN other words, if the contract had invoice 20 done last month, and there is no invoice entered for this month yet, we want to print 21 and say projected to indicate that the invoice hasn’t even been created yet for this month.

Drop a variable into the column for the Payment No and also a label. Set the label text and color as shown.

11.13 Step 12 - Set the name of the Report Objects using the Report Tree

Got to View > Toolbars > ReportData Tree and you should see the following screen. Your objects will have generic names like dbtext1 and variable1. To make it more manageable, change the names as shown here, matching them to the correct object.
This tree shows you all of the objects in your report and lets you set certain properties for those objects. We are going to have lots of variables for this report so we want to give them meaningful names rather than using generic names created automatically for us. Click on the variable object and rename it to vblInvoiceNo.
11.14 Step 13 - Add remaining variable objects

Just as with the invoice number variable, we have to do special handling for all of the following. Add variables for all of these:

vrblsIsCompleted
vrblsIsApproved
vrblsIsPaid
vrblsIsFinal
vrblsDaysLate
vrblsContractorPaid

Position them as shown here:

11.15 Step 14 - Add RAP code to handle the invoice number variable and visible properties

Now let's start adding our RAP code to set and use all of these variables. Go to the Calc tab and click on the Detail band. In the events window, highlight the BeforeGenerateEvent and insert all of this code:

```pascal
var
  CurrYear: integer;
  CurrMonth: integer;
  CurrDay: integer;
  ThruYear: integer;
  ThruMonth: integer;
  ThruDay: integer;
  ThruDate: Date;
begin
  ThruDate := CONTRACT['THRU'];
  DecodeDate(ThruDate, ThruYear, ThruMonth, ThruDay);
  DecodeDate(CurrentDate, CurrYear, CurrMonth, CurrDay);

  if (ThruMonth <> CurrMonth) or (ThruYear <> CurrYear) then begin
    vrblInvoiceNo.Value := CONTRACT['INV_NO'] + 1;
    lblProjected.Visible := true;
    vrblIsReceived.Visible := false;
    vrblIsCompleted.Visible := false;
    vrblIsApproved.Visible := false;
    vrblIsContractorPaid.Visible := false;
    vrblIsFinal.Visible := false;
    dbtFrom.Visible := false;
    dbtThru.Visible := false;
  end;
```
The code looks worse than it really is. It's just comparing the thru date of the invoice for a contract with the current system date. If it does not match the current month, then there is no invoice so it sets the vrblInvoiceNo variable to the next highest invoice and turns all of the other objects to invisible so they will not show.
11.16 Step 15 - Add RAP code for remaining variables

Now let's set the actual value for the remaining variables by writing a RAP event for each one.

Right click on the vrblsReceived variable and select calculations. An editor opens. Add the simple code as shown:

```plaintext
vrblsReceived:
if CONTRACT['INV_RECD_DATE'] = 0 then
    value := 'No'
else
    value := 'Yes';
```

repeat the same for all of the following using the code provided:

```plaintext
vrblsCompleted:
if CONTRACT['ENG93_COMPLETE'] = 'Y' then
    value := 'Yes'
else
    value := 'No';
```
vrblsApproved:
if CONTRACT['CEFMS_APPROVAL_STATUS']='A' then
  Value := 'Yes'
else
  Value := 'No';

vrblContractorPaid:
if RECEIPT_VOUCHER['V_AMT']>=CONTRACT['CURRENT_PAYMENT'] then
  Value := 'Yes'
else
  Value := 'No';

vrblsIsFinal:
if CONTRACT['FINAL']='Y' then
  Value := 'Yes'
else
  Value := 'No';

vrblDaysLate:
if RECEIPT_VOUCHER['V_AMT']>=CONTRACT['CURRENT_PAYMENT'] then
  DatePaid:=RECEIPT_VOUCHER['V_DATE']
else
  DatePaid:=RECEIPT_VOUCHER['V_DATE'];

if (CONTRACT['DUE_DATE']<>0) and (DatePaid<>0) and
  (CONTRACT['DUE_DATE']<DatePaid) then
  Value:=DatePaid-CONTRACT['DUE_DATE']
else
  Value:=0;

11.17 Summary

This report used a lot of RAP code and some interesting Oracle functions and groups. Its a good example of how a simple looking report can get complex due to the storage of the data.
RMS Report Writing

Example Report
For Training Purposes

Example XII
12 Example 12 - Scheduled Construction Events

12.1 Overview

This report shows some of the scheduled events for contracts. It includes milestones, preparatory inspections, initial inspections, correspondence and change requests. It uses many subreports all linked to one driving dataview and shows how to handle some advanced autosearch criteria issues.

Skills shown:
- Multiple SubReports
- Sharing Autosearch criteria in multiple queries.
- Subreport shift relative to.
- Accessing the text of autosearch criteria
- RAP Function to convert Dates to Strings

12.2 Step 1 - Load the Landscape Template and Set the title

Load the RMS Landscape Template and set the Report title
12.3 Step 2 - Create the Contract Query

From the data tab select File > New > Query Designer.

Select the CONTRACT table and add the following fields

Switch to the sort tab and sort the query by FMT_CONTRACT_DELIVERY
12.4 Step 3 - Create the EVENT Query

Select File > New > Query Designer

Select the EVENT table and select the following fields:
Switch to the sort tab and sort the table by DATE_SCH
Switch to the Search tab and set the following criteria:

![Query Designer](image)

**NOTE:** We will reuse this same criteria on all of the rest of our pipelines by using some RAP code. This is the only one we will set in the query designer. If we set all of the queries in the query designer, then the user would get prompted multiple times.

### 12.5 Step 4 - Create the Prep Query

Select File > New > Query Designer and select the FEATURE_QA table and then the FEATURE table. Join them by CONTRACT_ID and FEATURE_ID using an inner join.
Join Table

Join Type: Inner

FEATURE Fields:
- CONTRACT_ID
- FEATURE
- FEATURE_ID
- FINISHED
- FINISH_DATE
- FIRST_ACTDS
- FIRST_ACTID
- FOLLOW_COUNT
- GOV_STATUS

FEATURE_QA Fields:
- CONTRACT_ID
- FEATURE_ID
- INIT_HELD
- INIT_SCHL
- INIT_TIME
- PREP_HELD
- PREP_SCHL
- PREP_TIME
- QAVERSION

Joined Fields:
- FEATURE.CONTRACT_ID = FEATURE_QA.CONTRACT_ID
- FEATURE.FEATURE_ID = FEATURE_QA.FEATURE_ID

OK Cancel
Select the fields as shown here:

<table>
<thead>
<tr>
<th>Field</th>
<th>Alias</th>
<th>SQL Alias</th>
</tr>
</thead>
<tbody>
<tr>
<td>FEATURE_ID</td>
<td>FEATURE_ID</td>
<td>FEATURE_ID</td>
</tr>
<tr>
<td>INIT_TIME</td>
<td>INIT_TIME</td>
<td>INIT_TIME</td>
</tr>
<tr>
<td>CONTRACT_ID</td>
<td>CONTRACT_ID</td>
<td>CONTRACT_ID</td>
</tr>
<tr>
<td>CONTRACT_TYPE</td>
<td>CONTRACT_TYPE</td>
<td>CONTRACT_TYPE</td>
</tr>
<tr>
<td>PREP_TIME</td>
<td>PREP_TIME</td>
<td>PREP_TIME</td>
</tr>
</tbody>
</table>

And set the sort order as follows:
12.6 Step 5 - Create the Initial Query

Select File > New > Query Designer and select FEATURE_QA and FEATURE. Join them by CONTRACT_ID and FEATURE_ID as before using an inner join.
Set the fields as follows:

And the sort:
12.7 Step 6 - Create the SER_LOG query

Select File > New > Query Designer. Select SER_LOG from the table list and set the fields as shown here:
And the sort as shown here:
12.8 Step 7 - Create the PC_LOG query

(Last one folks)

Select File > New > Query Designer. Select the PC_LOG table and select the following fields:

<table>
<thead>
<tr>
<th>Field Name</th>
<th>SQL Alias</th>
<th>Table Alias</th>
</tr>
</thead>
<tbody>
<tr>
<td>AYS_DUE</td>
<td>SER_LOG.AYS_DUE</td>
<td>SER_LOG</td>
</tr>
</tbody>
</table>
And finally, sort it by MOD_DUE
12.9 Step 8 - Link all the Dataviews

Link each of the queries as shown here by dragging the CONTRACT_ID from the query into the CONTRACT_ID in the CONTRACT query.

Your screen should look something like this
12.10 Step 9 - Configure the queries to share autosearch values

This operation requires some voodoo code to get the search criteria from the EVENT query (the date range) and to have it apply to all of the other dataviews.

This code does the job. It basically adds 2 search criteria to each of the dataviews using the values from the autosearch entries on the event query.

```pascal
var
  lSQLBuilder: TdaSQLBuilder;
begin
  lSQLBuilder := TdaSQLBuilder.Create(FEATURE_QA);
  lSQLBuilder.SearchCriteria.Add('feature_qa', 'prep_schl', '>=', Report.AutoSearchFields[0].Value);
  lSQLBuilder.Free;

  lSQLBuilder := TdaSQLBuilder.Create(FEATURE_QA2);
  lSQLBuilder.SearchCriteria.Add('feature_qa', 'init_schl', '>=', Report.AutoSearchFields[0].Value);
  lSQLBuilder.Free;

  lSQLBuilder := TdaSQLBuilder.Create(SER_LOG);
  lSQLBuilder.SearchCriteria.Add('ser_log', 'ans_due', '>=', Report.AutoSearchFields[0].Value);
```
12.11 Step 10 - Add a subtitle to display the date range

Click on the design tab and add a label beneath the report title. Change its name to lblDateRange using the ReportTree.

Switch to the Calc tab again and highlight the report object. Add an event for the BeforePrint and set
the lblDateRangeText using this code:

```pascal
begin
  lblDateRange.Text := DateToStr(Report.AutoSearchFields[0].Value) + ' thru ' + DateToStr(Report.AutoSearchFields[1].Value);
end;
```

This will set the text to the date values that the user actually types in when the report runs. Note the need to convert the date into a string type so that the strings can be concatenated together.

12.12 Step 12 - Add a Contract group

Switch to the design tab and select Report > Groups and add a group as shown here for the CONTRACT_ID
Drag down the Group header to make room and format it as seen here with a background shape and DBText objects for the contract number and title:
12.13 Step 13 - Add 5 Subreports

Add five subreports and set the Datapipelines as shown here. To set the pipeline on each, click on the subreport in the tab at the bottom of the designer and then select Report > Data
12.14 Step 14 - Set the ShiftRelative to on all of the Subreports

If you don’t tell the subreports to shift relative to each other, then they will just print on top of each other. Right click on subreport 2 and set its ShiftRelative to SubReport1.

Do the same for the remaining subreports in sequence so the shift relative to each other from top to bottom.
12.15 Step 15 - Design all 5 subreports

Design each of the subreports as you see fit - Don't worry about the details - Just make a header and add the primary data. You can pretty them up later.

Events:
Example 12 - Scheduled Construction Events

**Preps:**

![Preparatory Inspection Diagram]

**Initials:**

![Initial Inspections Diagram]

**Correspondence:**

![Correspondence Requisitions Diagram]

**Change Requests:**

![Change Requests Diagram]
12.16 Summary

This report had 5 subreports and some fancy RAP code to get to the autosearch options and share it between queries. It also showed how to add a subtitle to show the range of data selected by the user. You could easily modify this report to add other tables or remove some of these that you are not interested in. Or, you could use a drill down option. See the Driller example for more info on that option.
13 Example 13 - Interactive Invoice Drilldown

13.1 Overview

Did you know reports can be clickable? This interesting example shows how you can turn a report into an interactive tool that lets the user expand and contract sections at run time. This report will list the invoices for a contract and let the user drill into the invoice and see CLIN payment details and further down to see activity payment details.

Skills shown:
- Interactive subreports
- Creating a customized view
- Multi-chained queries
- Custom Report Views
- Nested subreports

13.2 Step 1 - Load the standard Portrait Template

Load the standard portrait template with contract query and set the title as shown here.
13.3 Step 2 - Create the Request Query

Create a new query, select the REQUEST table and select the following fields
Set the sort to INV_NO as shown here
13.4 Step 3 - Create the CLIN query

For the CLIN query, we need to get the sum for the CLIN by adding up the activities linked to that CLIN. This involves linking to the ACTIVITY_PAY_HISTORY_TABLE and ACTIVITY_DESC tables in order to get the links we need. You could do this in the query designer, but lets use a view to make it easier.

Switch to the data tab and select File > Create View

When the View Browser comes up, click on Add View. Here is the SQL for the view we want to create.

```sql
SELECT
    B.CONTRACT_ID,
    AH.INV_NO,
    B.BID_NO,
    B.BID_DS,
    SUM(AH.ALLOW_AMT) V_AMT
FROM
    BID B,
    ACTIVITY_PAY_HISTORY AH,
    ACTIVITY_DESC A
WHERE
    A.CONTRACT_ID=B.CONTRACT_ID AND
    A.BID_NO=B.BID_NO AND
    AH.CONTRACT_ID=A.CONTRACT_ID AND
    AH.ID=A.ID
GROUP BY
    B.CONTRACT_ID,
    AH.INV_NO,
    B.BID_NO,
    B.BID_DS
```
After you have created and saved your view, select File > New > Query Designer and then select your new view RBVU_BID_PAY_HISTORY.

Add all of the fields to the selected list as shown here:
Sort the dataview by INV_NO and BID_NO so it will come out on the report in CLIN order.
13.5 Step 4 - Create the Activity Query

As with the BID query, it will be easier to get our activity history if we use a view.

Select File > Create View and create the following view for activity payment info

SELECT
    AH.CONTRACT_ID,
    AH.INV_NO,
    AH.ID,
    AH.ALLOW_AMT,
    AD.BID_NO,
    AD.DS
FROM
    ACTIVITY_PAY_HISTORY AH,
    ACTIVITY_DESC AD
WHERE
    AD.CONTRACT_ID=AH.CONTRACT_ID AND
    AD.ID=AH.ID
Now, add a dataview that uses the new view. Set the fields as shown here:

```
SELECT
    AH.CONTRACT_ID,
    AH.INV_NO,
    AH.ID,
    AH.ALLOC_AMT,
    AD.BID_NO,
    AD.DS
FROM
    ACTIVITY_PAY_HISTORY AH,
    ACTIVITY_DESC AD
WHERE
    AD.CONTRACT_ID = AH.CONTRACT_ID AND
    AD.ID = AH.ID
```
And the sort as shown here
RMS Report Writing
13.6 Step 5 - Link the queries

Now we need to link our four dataviews together in a chain like fashion. We want the REQUEST table to be linked to CONTRACT and the CLIN table to be linked to the REQUEST table and finally, the activity table to be linked to the CLIN table.

Set the links up as shown here:

Request to contract

![Link REQUEST as Detail](image)

Linked Fields

1. Include only those records from CONTRACT for which matching REQUEST records can be found.
2. Include all records from CONTRACT, regardless of whether any matching REQUEST records can be found.

Bid to Request
Activity to Bid
This is how the entire data scheme should look
13.7 Step 6 - Add Contract Group

Now that the data is set, let's design the report.

First, add a group for the contract. Switch to the designer and select Report > Groups and add a group as shown here:
Drag the group header downward with your mouse to make room and format a group header as shown here:
13.8 Step 7 - Add a SubReport

Now, we need to have a subreport for each section of our interactive report. One for the invoice data, one for the CLIN data, and one for the activity data.

Add a subreport for the Invoice data
13.9 Step 8 - Design the Invoice subreport

Switch to the invoice subreport by clicking on the tab at the bottom of the designer.

The first thing to do on any subreport is to set the data pipeline. Select Report > Data and set this subreport to use Request.

Now, design the subreport. We don't need a title band or footer so turn off everything else except the detail band (turn them off from the report pulldown menu).

Add DBText objects for the invoice Number, invoice date and amounts as shown.

Add a Label and change its color and make it underlined as shown so it looks like a clickable link and change the text.

Click on this hyperlink label so it is selected and then select View > Toolbars > ReportTree.
With the hyperlink label selected, right click on it in the tree and change its name to lblCLINDetail

Finally, add a subreport underneath the DBText objects - This will be the subreport for the CLIN details
13.10 Setp 9 - Configure the Invoice Subreport for Drilldown

Making a subreport interactive is very easy. Just right click on the Invoice subreport and select Drilldown so it is checked.

- Child
- Fixed
- Section

- Drill Down
  - Expand All
  - KeepTogether

- ParentWidth
  - Position...

- ShiftRelativeTo...

- Visible

When you make it DrillDown capable, it will ask you what control should act as the drilldown control. For this report, select lblCLINDetail which is our hyperlink looking label.
13.11 Setp 10 - Design the CLIN subreport

Switch to the CLIN subreport and set the data pipeline to RBVU_BIDPAY_HISTORY.

Design the subreport as shown here including adding the hyperlink type label and the Activity subreport.

Don’t forget to set the subreport to DrillDown and link it to the hyperlink label.

13.12 Setp 11 - Design the Activity subreport

Finally, design the activity subreport as shown here to complete the report.
13.13 Summary

This interesting report shows how simple it is to make a report interactive by enabling drilldown on a subreport. It also showed how to do nested subreports and how to chain together dataviews.
Example 14 - Payment Obligation Crosstab

14.1 Overview

If you have numeric data that gets sliced in 2 dimensions and you want to see the details, you can use a crosstab report component to build a report. This example shows a simple crosstab for the payments on a contract and how each payment is split across obligation lines.

Skills shown:
- Interactive subreports
- Creating a customized view
- Multi-chained queries
- Custom Report Views
- Nested subreports

14.2 Step 1 - Load the Landscape Template

Load the landscape template and set the title as shown here:
14.3 **Step 2 - Create the DataPipeline**

Switch to the data tab, select File > New > Query Designer

Select the PAY_OBLI_HISTORY table and add the fields as shown here.
For now, set the report to work with just one contract by hard coding a contract id into the search field.
14.4  **Step 3 - Add the crosstab component**

Drag the header down a little to make room for a crosstab control and then drop a crosstab control into the header band as shown here:

Shrink down the detail band as we will not have any details on this report.
14.5 Step 4 - Configure the Crosstab Control

Click on the crosstab and make sure the datasource is set in the edit bar.

Right click on the crosstab and select configure.
This is where you setup the rows, columns and values to display in the crosstab matrix.

Drag the INV_NO field from the left hand side and drop it onto the "new row" cell

Drag the LINE_ITEM field and drop it onto the "new column" cell

Drag the EARNINGS_US field onto the "new value" cell

That's all you have to do to configure the crosstab - the engine will take care of all of the horizontal totals, vertical totals and grand totals for you.

14.6 Step 5 - Format the Crosstab Items

While in the crosstab configuration, you can control some of the display formatting. Here are a few examples:

- Right click on the number in the cell and set the displayformat to a currency style:
- If you didn’t want grand totals for some reason, you can double click on the Grand Total label and they will be removed - Double Click again and they come back.

- You can change the labels, but you can change the fonts and colors

14.7 **Step 6 - Pagination**

Close the configuration screen and go back to the designer.

Right click on the crosstab and notice there are several options.

One important option is pagination. If your matrix is bigger than a page, then this will control how the pages print. There are two options

- Across then down
- Down then across

Across then down, or Down then across. These will control the behavior of the pages for big matrices.
14.8 Summary

It's actually very easy to build crosstabs using Report Builder as this demo has shown.
Example XV
15 Example 15 - Office Workload by Fund Type (Piechart)

15.1 Overview

This report shows how to include a chart and how to link it to a datapipeline. For this example, we will have a pie chart for each office and breakdown the sum of the contracts by Fund Type.

Skills shown:
- Using a chart
- Linking a chart to data in a pipeline
- Simple Subreport
- Simple Join

15.2 Step 1 - Load the Landscape Template

Load the standard Landscape template and change the title as shown here:
15.3 Step 2 - Create the Office Query

Switch to the data tab and select File > New > Query Designer

Select the offices table and select the following fields
15.4 **Step 3 - Create the Contract Query**

For the contract query, we want to sum up the current contract amount by office and program type.

Select File > New > Query Designer

Select the contract table from the list

We also want to get the description of the fund type so, next select the FUNDTYPE table and do an inner join.

Set the sort to OFFICE_SEQUENCE
Select the following fields
And add a calc to sum up the current contract amounts
And finally, group as follows:
15.5 Step 4 - Link the Contract Query

Set the linkage as shown here by dragging the office symbol from the contract query to the FLSYMBOL of the offices query.
15.6 Step 5 - Design the main report

Add DBText objects for the office symbol and the name as shown

Also, add a subreport for the actual chart - Put the subreport into the detail band:
15.7 Step 6 - Design the subreport and chart

Open the subreport (click on tab at bottom of designer) and condense down all of the sections except the title.

Set the datapipeline to the contract query.

Stretch out the Title band and Add a DBChart to the title band.

Right click on the dbChart and select Edit Chart
Click add series and select Piechart
Then click on Series on the top set of tabs
Example 15 - Office Workload by Fund Type (Piechart)

Click on the data subtab and set the datasource as shown here
15.8 Step 7 - Format the Chart

You can size and format the chart as needed. Here are a few examples:

- Under Panel you can select gradient, check the visible box and then pick an available gradient.
Under Series, General, you can control the display of the values and add commas or dollar signs as you like:
15.9 Summary

This example showed how to build a simple pie chart for offices and how to format and stylize the chart.
16 Example 16 - Project Data Sheet with Image

16.1 Overview

This report shows how you might put together a full page data sheet for a contract. It's based on a format from AED. We won't fill in every detail, but we will make it complete enough to serve as an excellent example for constructing your own data sheet. As you can see, one of the highlights is that it includes a photo which has been requested frequently by RMS users. This example will show how to get the photo out of the database and display it in a report.

Skills shown:
- Full page report
- Including an image in a report
16.2 Step 1 - Create the Contract Query

We need to get data from several tables for this report. They all need to reference back to the contract. We will not be loading a template for this report, so just switch to the data tab and select File > New > Query Designer

Select the following tables as shown here and join them as indicated. Note that they are all Left Outer joins so that we will still get a contract record if one of the other tables is missing a matching row.

Select the contract table:

Next, select the CR_ATTACH table - it is the table that stores attachments. We want our progress photo attachment so we will join the CR_ATTACH table by contract_id - Later we will limit it to just the progress photo

We need the contractor name, so next, select the KTR_DESC table and join it by contract_id also:
Next select the CONTRACT_METHOD table so we can get a description of the contract method.
Next up is the offices table - join it as shown here:

![Join Table](image)

We want the PM so join the PROJECT_DELIVERYTEAM table as shown
And, finally to get the full name of the PM instead of just the ID, join the staff table as shown here:

```
<table>
<thead>
<tr>
<th>STAFF Field</th>
<th>Operator</th>
<th>PROJECT_DELIVERY_TEAM Field</th>
</tr>
</thead>
<tbody>
<tr>
<td>STAFF.EMP_ID_NO</td>
<td>=</td>
<td>PROJECT_DELIVERY_TEAM.EMP...</td>
</tr>
</tbody>
</table>
```
The tables and joins should now look like this:

![Query Designer](image)

Select the following fields:

<table>
<thead>
<tr>
<th>Table</th>
<th>Table Name</th>
<th>SQL Alias</th>
<th>Join Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>CONTRACT</td>
<td>CONTRACT</td>
<td>CONTRACT</td>
<td></td>
</tr>
<tr>
<td>CP_Attach</td>
<td>CP_Attach</td>
<td>CP_Attach</td>
<td>Left Outer</td>
</tr>
<tr>
<td>KTR_DESC</td>
<td>KTR_DESC</td>
<td>KTR_DESC</td>
<td>Left Outer</td>
</tr>
<tr>
<td>CONTRACT_ID</td>
<td>CONTRACT_ID</td>
<td>CONTRACT_ID</td>
<td>Left Outer</td>
</tr>
<tr>
<td>OFFICES</td>
<td>OFFICES</td>
<td>OFFICES</td>
<td>Left Outer</td>
</tr>
<tr>
<td>PROJECT_DEU</td>
<td>PROJECT_DEU</td>
<td>PROJECT_DEU</td>
<td>Left Outer</td>
</tr>
<tr>
<td>STAFF</td>
<td>STAFF</td>
<td>STAFF</td>
<td>Left Outer</td>
</tr>
</tbody>
</table>
And set these search conditions:
The contract_id is just so we can work with a single contract while we are designing (pick a contract_id of your own choosing)

The CR_ATTACHMENT.DOC_TYPE of J is the indication of a progress photo (jpeg)

The KTR_DESC.P_S of type P is so we get the name of the Prime contractor

And, the TITLE of Project Manager is self explanatory. Notice, however, that you need to add the outer join condition (+) to the TITLE fields here or you will filter out any contracts that do not have a PM assigned.

You could sort the data in any way you choose, but we will just work on one contract.
16.3 **Step 2 - Create the Office Query**

We want the district name in big bold letters at the top of the report so we need to go get it from the office table.

Close your contract query and save it. Then, select File > New > Query Designer

Select the office fullname.
Finally, set the search to just the district office by checking the type_value.
You should now have 2 dataviews as shown here - Note - they are not linked and that is the way we want it.
16.4  **Step 3 - Design the Report**

Here is what the final design should look like in the designer. There are a lot of steps involved, but they are all basic. The only new thing is the Image. Design your report to look just like the one shown here. Here are a few highlighted steps:
- Get rid of everything except the header and detail section - we don't need a footer, summary or title.

- Use regions to control the flow of the report. In the design as shown, there are two regions dividing the detail band in half for the upper data and then a new region for each of the lower sections.

- The DBImage (ATTACHMENT) is easy to setup, just grab the DBImage control and drop it into the region and size it. Set it to use the CONTRACT pipeline and the ATTACHMENT field. Also, right click on it and set the graphic type to JPEG.

- The small header blocks like PROJECT, CONTRACT, FUNDING are just regular labels with a blue background. You can set the background by using the View > Toolbars > Report Data Tree and setting the Color under Appearance to skyBlue.

- The Variable 1 and Variable 2 at the top of the page each have the following code set for them. You access the code editor by right clicking on the variable and selecting calculations from the menu.
Make sure to set the Stretch and Shift Relative To on the Lower regions and the memos so that they will stretch correctly.

16.5 Summary

This rapid fire example showed how you can make up your own data sheet and how you can include an image. You can easily select whatever fields you want to include and modify it for the needs of your office.
17 Example 17 - Contractor Evaluation Grades

17.1 Overview

This advanced report uses lots of RAP code including user defined functions and global variables. The report finds all of the contracts for a given contractor by using the DUNS number. It summarizes and averages the ratings from all of the evals for the given contractor.

Skills shown:
- Advanced RAP
- User defined functions
- Global variables
17.2 Step 1 - Load the Portrait Template

Load the Standard Portrait Template and change the Title of the Report

17.3 Step 2 - Create a View of DUNS numbers

Switch to the data tab and select File > Create View

- New...
- Close...
- Import...
- Merge...
- Export...
- Import From File...
- Export To File...
- Launch ISQL

Select Create View

- Save Report With View To File
- Load Report With View From File

On the View Edit dialog, enter the SQL statement as shown.

select
DISTINCT(DUNS4) DUNS
17.4 Step 3 - Create the DUNS Query

Select File > New > Query Designer and then select the RBVU_DISTINCT_KTR_DUNS4 table
Select the DUNS field

### Available Tables

<table>
<thead>
<tr>
<th>Table</th>
<th>Table Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>ACCIDENT</td>
<td>ACCIDENT</td>
</tr>
<tr>
<td>ACTIVITY_DESC</td>
<td>ACTIVITY_DESC</td>
</tr>
<tr>
<td>ACTIVITY_PAY_HI</td>
<td>ACTIVITY_PAY_HI</td>
</tr>
<tr>
<td>ACTIVITY_QA</td>
<td>ACTIVITY_QA</td>
</tr>
<tr>
<td>ACTIVITY_SCHED</td>
<td>ACTIVITY_SCHED</td>
</tr>
<tr>
<td>AGENCY</td>
<td>AGENCY</td>
</tr>
<tr>
<td>AGENT</td>
<td>AGENT</td>
</tr>
<tr>
<td>AHA_HAZARD</td>
<td>AHA_HAZARD</td>
</tr>
<tr>
<td>AHA_STEP</td>
<td>AHA_STEP</td>
</tr>
<tr>
<td>ANT_WEATHER</td>
<td>ANT_WEATHER</td>
</tr>
<tr>
<td>APPVU_CONTRACT</td>
<td>APPVU_CONTRACT</td>
</tr>
<tr>
<td>APPVU_CONTRACT</td>
<td>APPVU_CONTRACT</td>
</tr>
<tr>
<td>APPVU_CONTRACT</td>
<td>APPVU_CONTRACT</td>
</tr>
<tr>
<td>BASELINE_LIBRAR</td>
<td>BASELINE_LIBRAR</td>
</tr>
<tr>
<td>BASELINE_OFFICE</td>
<td>BASELINE_OFFICE</td>
</tr>
<tr>
<td>BID</td>
<td>BID</td>
</tr>
<tr>
<td>BID_CHANGE</td>
<td>BID_CHANGE</td>
</tr>
<tr>
<td>BID_CHANGE_OBL</td>
<td>BID_CHANGE_OBL</td>
</tr>
</tbody>
</table>

### Selected Tables

<table>
<thead>
<tr>
<th>Table</th>
<th>Table Name</th>
<th>SQL Alias</th>
<th>Join Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>RRVU_DISTINCT_KTR_DUNS</td>
<td>RRVU_DISTINCT</td>
<td>RRVU_DISTINCT</td>
<td></td>
</tr>
</tbody>
</table>
For testing/design purposes, set the DUNS to get only the single DUNS as shown here on the search tab.
17.5 Step 4 - Create the Eval Query

Select File > New > Query Designer and then select the FORM2626 table. This is the table that has the Evaluation data.
Select the following fields
We need to get the location, so switch to the calc tab and add a calculated field using this expression:

\[
\text{(SELECT CONTRACT.LOCATION FROM CONTRACT CONTRACT WHERE CONTRACT.CONTRACT\_ID = FORM2626.CONTRACT\_ID)}
\]
Also, we only want the last evaluation. To handle that, enter this expression into the search criteria:

(select MAX(EVAL.EVA_NO) from form2626 EVAL where EVAL.CONTRACT_ID = FORM2626.CONTRACT_ID)
Finally set the sort order as shown here:
17.6 Step 5 - Create the Eval Global Query

Now we will create a global eval query that we will use to do the actual calculations for the ratings and averages.

Select File > New > Query Designer and select the FORM2626 table again
Select the following fields

<table>
<thead>
<tr>
<th>Table</th>
<th>Table Name</th>
<th>SQLAlias</th>
<th>Join Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>FORM2626</td>
<td>FORM2626</td>
<td>FORM2626</td>
<td></td>
</tr>
</tbody>
</table>
Set the search criteria as shown here using this oracle expression.

\[
\text{(select MAX(EVAL.EVA_NO) from form2626 EVAL where EVAL.CONTRACT_ID = FORM2626.CONTRACT_ID)}
\]

Finally, sort by DUNS
17.7 Step 5 - Link the FORM2626 to the RBVU_DISTINCT_KTR_DUNS4 query

Link the two dataviews as shown here by dragging the DUNS4 field from the FORM2626 to the RBVU_DISTINCT_KTR_DUNS4 query.
Set the link settings as shown:

1. Include only those records from RBVU_DISTINCT_KTR_DUNS4 for which matching FORM2626 records can be found.
2. Include all records from RBVU_DISTINCT_KTR_DUNS4, regardless of whether any matching FORM2626 records can be found.
17.8 Step 6 - Design the Report

On the design tab, layout your report as shown here.
17.9 Step 7 - Refer to the finished example for RAP details

For learning purposes, refer to the completed example for all of the RAP coding that drives the report. Explore the various coded events and global functions for some good examples on how to handle some advanced RAP concepts.
18 Example 18 - RMS Construction Calendar

Enter topic text here.

18.1 Overview

This report shows how to create a calendar of events. It uses an advanced SQL statement to create a view with a row for every possible day. That view becomes the main query in the report and then all other queries for actual events are simply linked to it.

Skills shown:
- Advanced SQL (SQL unions)
- User create view

18.2 Step 1 - Add a view to create daily records

We will use some crossjoins to create numerous daily records within a view since we don't have a table in RMS for every day available.

Here is the SQL:

```
select trunc(sysdate-10000 + digit5.val + digit4.val + digit3.val + digit2.val + digit1.val) as v_date from
(select 0 val from dual union
```
select 1 val from dual union
select 2 val from dual union
select 3 val from dual union
select 4 val from dual union
select 5 val from dual union
select 6 val from dual union
select 7 val from dual union
select 8 val from dual union
select 9 val from dual)
digit1
cross join
(select 0 val from dual union
select 10 val from dual union
select 20 val from dual union
select 30 val from dual union
select 40 val from dual union
select 50 val from dual union
select 60 val from dual union
select 70 val from dual union
select 80 val from dual union
select 90 val from dual)
digit2
cross join
(select 0 val from dual union
select 100 val from dual union
select 200 val from dual union
select 300 val from dual union
select 400 val from dual union
select 500 val from dual union
select 600 val from dual union
select 700 val from dual union
select 800 val from dual union
select 900 val from dual)
digit3
cross join
(select 0 val from dual union
select 1000 val from dual union
select 2000 val from dual union
select 3000 val from dual union
select 4000 val from dual union
select 5000 val from dual union
select 6000 val from dual union
select 7000 val from dual union
select 8000 val from dual union
select 9000 val from dual)
digit4
cross join
(select 0 val from dual union
select 10000 val from dual union
select 20000 val from dual union
select 30000 val from dual)
digit5
order by v_date
18.3 Step 2 - Configure the Queries

We will have six queries. The main one will use our view for the days.

Add a query for the days:

```sql
SELECT TRUNC(SYSDATE - 10000 + D.I + D.II) AS DAY FROM D;
(SELECT 0 VAL FROM D) UNION
(SELECT 1 VAL FROM D) UNION
(SELECT 2 VAL FROM D) UNION
(SELECT 3 VAL FROM D) UNION
(SELECT 4 VAL FROM D) UNION
(SELECT 5 VAL FROM D) UNION
(SELECT 6 VAL FROM D) UNION
(SELECT 7 VAL FROM D) UNION
(SELECT 8 VAL FROM D) UNION
(SELECT 9 VAL FROM D) CROSS JOIN
(SELECT 0 VAL FROM D) UNION
(SELECT 10 VAL FROM D) UNION
(SELECT 20 VAL FROM D) UNION
(SELECT 30 VAL FROM D)
```
Select all fields

Add an activity schedule query that joins to contract:
Select these fields
Add a request query that links to contract as shown here
And select these fields
Add an event query that links to contract as shown:
And select these fields
Add a feature_qa query that links to contract for the prep dates
And select these fields:
And for the last query, add a second query for FEATURE_QA for the initial inspections
Using these fields:
18.4 Step 3 - Link the queries using the dates

Link the queries as shown here using the date fields from each.
18.5 Step 4 - Design the Report

Design the report using a region that contains subreports as shown here:
Set the two dbtext objects at the top of the band to point to the V_DATE field. Set the display format of the first one to dddd which will print the day of the week and set the second one to mmmm d, yyyy which will display the spelled out date of the year.

18.6 Step 5 - Design the Subreports

Design each subreport as shown. Make sure to configure the shiftrelative to values. You may have to use the report tree to change the shiftrelative to since the subreports are within a region.
19  Example 19 - Mail Merge

19.1  Overview

This example gives a quick demonstration of how the mailmerge features work through the RichEdit Controls. This will allow you do do any type of form letter while still using all of the features of the report designer such as subreports. The example shows how to mix a subreport in between two rich edit controls.

**Contract No:** W9128A-06-D-0009 0001  
**Title:** FY06 OMA Pkg A-022 Repair Bldg 405

This contract was awarded on 9/29/2006 for the award contract amount of $1,478,279.00.

The scope of work for said contract is:

- Renovation of Bldg 405 includes the design, description, replacement or upgrade of offices, sections, etc. The renovation upgrades require compliance with current building codes and standards. The scope of work includes the replacement of the interior portion of the building plus inside HVAC and interior electrical systems. Exterior work includes the painting and replacement of windows and doors, the replacement of an entrance with a sidewalk, the installation of the AP Reader, and the installation of glassed areas.

This contract currently has the following 23 features of work:

For the RMS center reference contract IDJ3060861

- Award amount is $1,478,279.00
  - Acoustical Treatment
  - Ceramic Tile Flooring
  - Demolition
  - Door Hardware
  - Electrical
  - Fire Detection/Alarm
  - Fire Sprinkling
  - Glass/Glazing
  - Interior Doors

Skills shown:
- Basic Report Skills
- Using RichText controls
- How to configure and use mailmerge tags

19.2  Step 1 - Configure a Contract Query

This report will not use a standard template. To begin, just create a query for the contract table as shown here
Get the following fields
Add a calculation using an expression to get the number of features. We will create a mail merge tag for it later. Name it and use the expression as shown here:
19.3 Step 2 - Configure a Feature Query

For demonstration, we will include a list of features of work in the report. Configure a second query using the feature table.
Use these Fields
And sort by the feature title as shown here
Finally link the two queries as shown here:
19.4 Step 3 - Configure the Design with Richtext and Subreport controls

Switch to the design tab and put a richtext control at the top of the design band, then a subreport, then another richtext control as shown here. Turn off all of the other bands.

Add a group (Report > Groups) by contract ID and set it to start on a new page. This will make each form letter begin a new page.
19.5 **Step 4 - Configure the Richtext controls for mailmerge, and stretch and setup the shift relative to**

Right click on each of the Richtext controls and turn on mailmerge. Also be sure to set stretch on and finally configure the three design components to shoft relatice to each other appropriately.

19.6 **Step 5 - Add Some Mail Merge Text**

Right click on the first richtext control and select Edit. The richtext editor with mailmerge appears. Enter some text as shown below to try out the mail merge options. Note that you can use the Fields list on the right along with the displayformat and then click on add and the appropriate tag will be added for you. You can also manually type the field name in with brackets.

You can use an valid richtext formatting including indenting, font size, bold, italics and color
19.7 Step 6 - Configure the subreport

Configure the subreport to list out the features as shown here:

Make sure to set the data pipeline to FEATURE
19.8 Step 7 - Add some RichText for the summary section

Finally, to show how the subreport can be embedded in the middle of the formletter, add some richtext for the second richtext control
19.9 Summary

This example shows how to use richtext and mailmerge. The mailmerge is a powerful feature that gives you access to all fields. Also you can still mix in all of the other report designer options like subreports.
Back Cover