



Check Processing, Electronic Refund,
Document Enhancement, Web Enabled Reporting

Argos Basic Training - DataBlock Designer Workbook

Document Version 3.11

Last Updated 10.3.2013

Table of Contents

Introduction	4
Before We Begin: Find Your Logo.....	4
Glossary of Useful Terms.....	4
Important Note About This Workbook.....	5
First Exercise: Basic DataBlock 1 – Address List for Admitted Students	6
Exercise Description	6
Instructions.....	7
Create a new DataBlock.....	7
Open DataBlock for editing.....	7
Change the background color of the main form.....	7
Add a form title linked to the DataBlock’s name.....	7
Test that the label shows the name of the DataBlock.....	7
Add your school logo and a hyperlink to your school’s website.....	8
Add a new panel.....	8
Add a form description to the panel.....	9
Add the ‘Admission Type’ variable	9
Add the ‘Term’ variable.....	11
Add the ‘Address Type’ variable.....	13
Create a Report Query for this DataBlock.....	14
Supplemental Exercise: Create a DataBlock Template.....	19
Make a copy of the DataBlock.....	19
Edit the DataBlock to remove objects you don’t want in the template.....	19
Add remaining objects to the Library of Objects to create a template.....	19
Second Exercise: Basic DataBlock 2 – Budget Availability.....	21
Exercise Description	21
Instructions.....	22
Create a new DataBlock and open it for editing	22
Configure the form layout and add template objects.....	22
Add the ‘Chart of Accounts’ variable	22
Add the ‘Fiscal Year’ variable.....	24
Add the ‘Fund’ variable.....	25
Add the ‘Organization’ variable.....	27
Add ‘Account’ variable	28
Add the ‘Run Dashboard’ button.....	29
Align the position of your objects	30
Create the multi-column list box to display budget information.....	30
Modify the column headings in the multi-column list box	32
Create Report Query for DataBlock by copying and modifying the multi-column list box query.....	33

Third Exercise: Basic DataBlock 3 – Student Course List.....	36
Exercise Description	36
Instructions.....	37
Create a new DataBlock and open it for editing	37
Configure the form layout and add template objects.....	37
Add the 'Term' variable.....	37
Add a sub-heading for the next two variables.....	39
Add the 'Last Name' variable.....	39
Add the 'ID Number' variable.....	40
Add the 'Get List of Students' button.....	40
Add a sub-heading for the multi-column list box.....	40
Add the 'Student List' multi-column list box to display student information.....	40
Modify the column headings in the multi-column list box	42
Add the 'Dropped Classes' checkbox.....	42
Create a Report Query for this DataBlock.....	43
Appendix I: DataBlock Naming Conventions.....	45
Naming Convention for Form Variables.....	45
Naming Convention for SQL Variables:	45

Introduction

Welcome to DataBlock Designer Training

Welcome to the DataBlock Designer training sessions for Argos. As you move through the exercises that follow, you will learn the basics of creating, formatting, and building DataBlocks that can be used by any Argos end user to view institutional data and to run reports.

The exercises that follow will give you hands-on training with many features of the DataBlock designer tool, and they will also will give you experience creating a variety of different types of forms and report queries from scratch.

Before We Begin: Find Your Logo

For the first exercise, you're going to need a copy of your school's logo, which you will include in your DataBlock design. Please find a copy of your school's logo (you can simply download the image from your website). Make sure you have the file available to you on your workstation.

Glossary of Useful Terms

Here are definitions for some common terms we'll be using in this training:

DataBlock – DataBlocks are containers that hold SQL (Structured Query Language) queries, which return data from your institution's data sources. When you create a new DataBlock, you have an empty container. You can then customize that container to include SQL statements, forms, variables, charts, graphs, etc.—anything you need to constrain and display the data requested. DataBlocks are the foundation from which all reports are created, and they allow you to organize SQL queries so it's easy for end users to create multiple reports using the same data sets.

Dashboard – The dashboard consists of all of the DataBlock elements that are visible to the end user. Any forms, variables, control objects, charts, graphs, OLAP (Online Analytical Processing) cubes, etc. that will be displayed to the DataBlock's end user are part of the dashboard. The dashboard is the end user's control panel for the DataBlock, where they can input their choices, view the resulting data set, and extract it as a report.

Form – Forms are the visible "pages" that users see when using a dashboard. A DataBlock can have one form or many. They start out as a blank canvas, to which you can add text, graphics, and many types of control objects. Oftentimes, when a DataBlock has more than one form, the end user will move back and forth amongst them like the different pages of a website.

Variable – A variable is an element that is used to constrain the data returned by the SQL queries in your DataBlock. When designing a DataBlock, you will often be adding variables to your DataBlock's form. In this case, the term 'variable' refers to the control object—like drop downs, text boxes, radio buttons, etc.— used by the end users to select values for the variables. The values they choose are then used to constrain the data set the SQL query will return.

Control Objects – Although control objects are often used to allow the end user to select variables, they can also be used for other purposes. Multi-column list boxes, for example, are used to display the data set currently being returned by the DataBlock, and buttons can be used to execute commands. You'll learn more about the different types of control objects as you progress through the exercises.

Report Query – In addition to adding visible elements to your DataBlock, you will also write report queries, which are the SQL queries used to generate reports. The report query gathers the data set that the end user will use when constructing a report. There can only be one report query per DataBlock.

Report – A report is simply a design structure, allowing a report writer to decide how they're going to display the data returned by the report query. Argos has three types of reports: Banded reports include graphics, charts and rich formatting. Comma-Delimited reports export the raw data set, allowing users to import it into other programs. Extract Text reports are also a basic report, but with more options than a Comma-Delimited report.

Important Note About This Workbook

Please note that this workbook is not designed to replace the user documentation—it simply provides step-by-step instructions you can follow along with for these training exercises. If, at any point, you'd like more details on a particular feature or are curious about a feature that the exercise does not cover, please refer to the in-product help.

First Exercise: Basic DataBlock 1 – Address List for Admitted Students

Basic Datablock 1 - Address List for Admitted Students

argos
powered by evisions

Find Addresses for Admitted Students

Select Admission Acceptance Type

- Early Admission
- Early Decision
- Special Program
- Standard

Select Term:

200940 - Summer 2 2009

Select Address Type:

- Billing
- Business
- Corporate Headquarters
- Corporate Subsidiary
- Emergency Contact
- Firm Address
- Mailing
- Matching Gift Address

Products Training Support Feedback

www.evisions.com

Exercise Description

This DataBlock returns a list of addresses for admitted students. Above is an example of what the finished DataBlock will look like at the end of the exercise.

In addition to learning how to create and format new DataBlocks, this exercise will also demonstrate how to add variables to a form. Here is a description of each of the three variables you'll be using for this DataBlock, and note on which database table they draw from:

Admission Type – list box variable that returns a list of admission type *Table used: STVADMT*

Term – drop down variable that returns a list of terms *Table used: STVTERM*

Address type – drop down variable that returns a list of address types. *Table used: STVATYP*

In addition to building a form with these variables, you will also build a Report Query, which will enable the end user to run a report of the dataset returned by the variables on the form.

'Address List for Admitted Students' Report Query – contains the SQL query to return student name, address, and admission type using the constraints imposed by the variable selection of Admissions Type, Term and Address Type. *Tables used: SPRIDEN, SPRADDR, SARADAP, STVADMT, STVAPST*

Instructions

Create a new DataBlock

1. Find your user folder in the **Explorer** tab folder list on the left-hand side.
2. Right-click on your folder, and choose **New> DataBlock** from the menu.
3. Type in the name of your new DataBlock – ‘Address List for Admitted Students’ – and hit the **Enter** key.
4. Click on your new DataBlock to highlight it.
5. On the right side of the screen, under the **DataBlock Designer Actions** heading, select the ADO being used for your training from the Associated Connection/Pool drop down.
NOTE: If you're not sure which ADO you're using, ask your trainer.

Open DataBlock for editing

6. Click to highlight your DataBlock.
7. On the right side of the screen, under the **DataBlock Designer Actions** heading, click the **Edit** button.

Change the background color of the main form

8. Make sure the **Form Design** tab in the upper left corner of the window is selected.
9. Under the **Form Design** tab, select the **Properties** tab.
NOTE: You'll be spending a lot of time changing things on the Properties tab—please refer to the 'DataBlocks > Form Design (Visual Designer)' section of the Argos in-product help for more information about the Properties tab.
10. Where it says **Color**, select the color you want from the drop down.

Add a form title linked to the DataBlock's name

11. Click on the **Add Label** tool (the letter A button on the toolbar).



12. Click on the form approximately where you want the DataBlock title to appear, which will create a textbox for the label.
13. On the Properties tab on the left-hand side, where it says **Data Aware**, select ‘Yes’ from the drop down.
14. On the same tab, where it says **Text**, instead of typing the DataBlock name, choose **\$DataBlock.Name**.
NOTE: By making the label data aware, it will ensure that any edits you make to the name of the DataBlock are reflected in this label, as well as allow you to duplicate the label in a DataBlock template.
15. You can also modify the appearance of the label from the **Properties** tab:
16. Make sure the label is still selected.
 - a. Next to **Font.Bold** choose ‘Yes’.
 - b. Next to **Font.Name** choose ‘Arial’.
 - c. Next to **Font.Size** type in ‘22’.
17. When you've made all your changes, click and drag your new label to upper left-hand corner of the form.

You'll notice that the label does not automatically display the name of the DataBlock but instead shows '\$DataBlock.Name,' the name of the variable the label is linked to. To make sure that it's connecting properly, you'll want to commit your changes and test the DataBlock.

NOTE: After any major changes you make to a DataBlock, it's a good idea to commit (i.e., save) your changes and test that everything you've done so far is working correctly.

Test that the label shows the name of the DataBlock

18. At the top of the window, click the **Commit** button to save.

19. To the right, click the **Test** button.
20. A preview window will launch and should display the name of your DataBlock.
21. Click the **Close** button when done to return to the DataBlock designer.

NOTE: You can test your DataBlock at any point—whether you’ve committed your changes or not—by following the directions above.

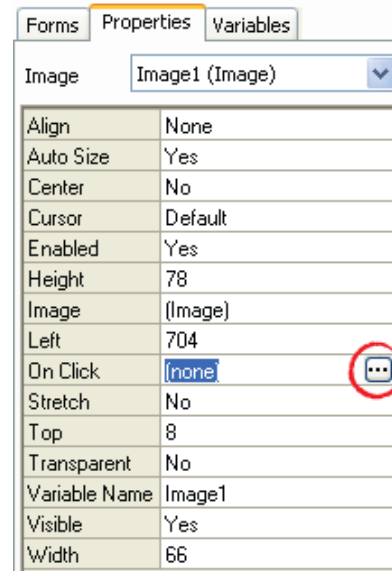
Now we know that the title’s working properly, we can start adding more elements to our form.

Add your school logo and a hyperlink to your school’s website

22. Add your school logo to your form.
 - a. Select **Add Image** tool (the picture frame button on the toolbar).



- b. Click anywhere on the form, and the **Image** window will appear:
 - i. Leave the default 'Image is stored in the DataBlock' option selected.
 - ii. Click on the **Browse** button.
 - iii. Browse to the location of your school’s logo and select it.
 - iv. Click **OK** to close the dialog.
 - c. On the **Properties** tab, next to **Auto Size** select 'Yes', to fit the bounding box to the size of the image.
 - d. Click and drag your logo to the upper right-hand corner of form.



23. Make the logo link to your school’s homepage by giving it an On Click action.
 - a. On the **Properties** tab, next to **On Click**, click where it says '(none)' and then click the ellipsis button that appears (see right) to open the **Edit Event** window:
 - i. Under **Available Events**, click on 'Hyperlink' and then click the **blue arrow** button to move it to the Event List.
NOTE: You can also double-click on an event to move it to the Event List.
 - ii. Under **Event Properties** type the URL for your school’s website into the **URL** text box.
 - iii. Click **OK** to close the window.
 - b. On the **Properties** tab, next to **Cursor** select 'Handpoint' from the drop down. This will make the cursor change to a hand when it rolls over the logo.
24. Commit and test your changes.
 - a. Click the **Commit** button to save.
 - b. Click the **Test** button to preview the form.
 - c. Test the changes you just made (in this case, that the link works and the cursor changes appearance).
 - d. Click the **Close** button when done.

NOTE: From now on, when you see “Commit and test your changes.” in these instructions, follow the four preceding steps.

Add a new panel

25. Select **Add Panel** tool (the blank square button on the toolbar).



26. Click on the form to create a new panel.
27. Click and drag one of panel's corners to increase its size until it fills most of the area beneath the DataBlock Name and your school logo.
28. Change the panel appearance in the **Properties** tab:
 - a. Next to **Color**, choose 'white'.
 - b. Next to **Font.Name**, choose 'Arial'.
 - c. Next to **Font.Size**, type '10'.

Add a form description to the panel

29. Click on the **Add Label** tool.
30. Click in the panel to place your new label within the panel.
NOTE: From now on, when you see "Add a new label to the panel." in these instructions follow the two preceding steps.
31. On the **Properties** tab:
 - a. Next to **Font.Bold**, choose 'Yes'.
 - b. Next to **Font.Size**, type '14'.
32. Double-click on the label object to open the **Modify Text** window:
 - a. Type 'Find Addresses for Admitted Students.'
 - b. Click the **OK** button.
NOTE: When you've finished the form, you can also come back and add more text here, to give users instructions for how to use the form.
33. Click and drag the description to the upper left-hand corner of the panel.

Add the 'Admission Type' variable

34. Add a label for your variable.
 - a. Add a new label to the panel. (Follow steps 29-30.)
 - b. Double-click and change the text to 'Select Admission Acceptance Type.'
 - c. Click and drag the label to the left side of the panel, under the description.
35. Add a list box control object.
 - a. Click on the **Add List Box** tool (the button on the toolbar that shows a list with a scroll bar).



- b. Click in the panel to place the list box.
 - c. On the **Properties** tab:
 - i. Next to **Variable Name**, type 'main_LB_AcceptCode'
NOTE: It's important to use a common naming convention when naming different elements of a DataBlock. For the purposes of this training we are going to use the Evisions naming convention. The names for form variables adhere to the following convention:

$$[FormName]_[AbbreviatedControlType]_[VariableDescription]$$
For more on naming conventions, please refer to Appendix I: Naming Conventions at the end of this workbook.
 - ii. Next to **Multi Select**, choose 'Yes'.
 - d. Click and drag the list box below your 'Select Admission Acceptance Type' label.
36. Write a SQL query to return a list of Admission Type codes for the list box.

- a. On the **Properties** tab, next to **Choices**, click where it says '(Choices)' and then click the ellipsis button that appears to open the **Choice Entry** window:

NOTE: You can also open the Choice Entry window by double-clicking on the list box.

- i. Select the **SQL Statement** radio button.
- ii. Click the **Next** button.
- iii. On the next screen, click on the **Edit Visual Design** button (the hardhat and hammer button on the toolbar) to open the **Build Query** window.



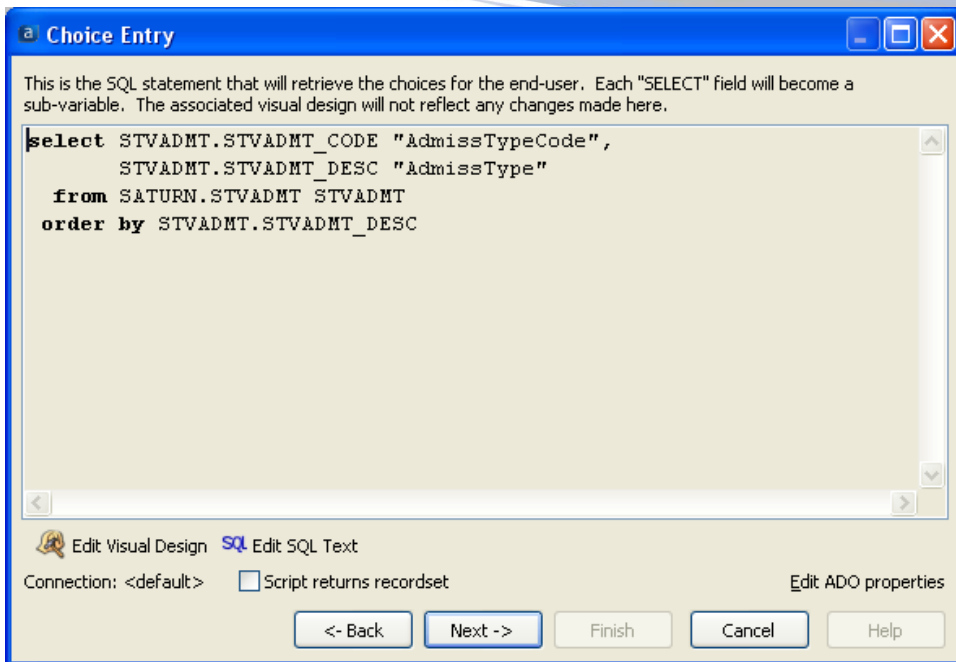
NOTE: The Visual Designer is very useful when writing SQL queries. Please refer to the DataBlocks > SQL Editor > Build Query section of the Argos in-product help for more information about building SQL queries with the Visual Designer.

- (1) In the **Build Query** window, click on the **Add Table** button at the top.
 - (a) Type in the name of the table you're looking for, in this case 'STVADMT,' and click the **OK** button.
 - (b) Select the 'STVADMT' table from the **Choose Table** window.

NOTE: There may be more than one selection with the same name—choose the one that says 'TABLE' beneath the Type heading (not the one that says 'SYNONYM').
 - (c) Click the **OK** button.
 - (2) The table you selected will appear in the **Build Query** window. At the bottom of the window, make sure the **Visible Fields (SELECT)** tab is selected.
 - (3) To tell your query which fields to select from, you want to choose fields from the 'STVADMT' table and add them to the **Visible Fields (SELECT)** tab. To add a field, double-click on the name of the field in the 'STVADMT' table window:
 - (a) Double-click on 'STVADMT_CODE'.
 - (b) Double-click on 'STVADMT_DESC'.


NOTE: Those two fields should now appear in the Visible Fields (SELECT) tab.
 - (4) You can add fields to the other tabs in the same way. For this query, you also want to add a field to the **Ordering (ORDER BY)** tab, which determines how the query results are sorted:
 - (a) Click the **Ordering (ORDER BY)** tab to select it.
 - (b) Double-click on 'STVADMT_DESC' in the 'STVADMT' table to add it to the **Ordering (ORDER BY)** tab.
 - (5) Click the **OK** button to exit the Visual Designer.

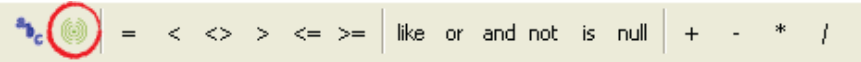
NOTE: Be careful not to click the Cancel button—you will lose all your work.
- iv. Review the SQL query you just built. It should look like this:



- v. Click the **Next** button, at which point Argos will validate the SQL query, checking for grammatical correctness. *NOTE: If there is an error in your query, an error message will display. If you have an error message:*
 - (a) Read the error message, looking for indications of what might be wrong.
 - (b) Click the **OK** button.
 - (c) Click the **Edit Visual Design** button to go back to the **Build Query** window to find and fix your error. Note that clicking **Cancel** will cancel all your work.
 - vi. Provided there are no errors, you should see a list of admission type codes and their descriptions. Click the **Next** button to continue.
 - vii. On the next page, select 'AdmissType' from the drop down. This will display the spelled-out admission type descriptions in the list, instead of the 2-letter codes (which would be less helpful to the end user).
 - viii. Click the **Finish** button.
- b. Commit and test your changes. (Follow steps 24a-d.)

Add the 'Term' variable

37. Add a label for your variable.
 - a. Add a new label to the panel. (Follow steps 29-30.)
 - b. Double-click and change the text to 'Select Term:'.
 - c. Click and drag the label to the right side of the panel, under the logo.
38. Add a drop down control object.
 - a. Click the **Add Drop Down** tool (button in the toolbar that looks like a drop down menu).
 
 - b. Click on the panel to place the drop down.
 - c. On the **Properties** tab:
 - i. Next to **Variable Name**, type 'main_DD_Term'.
 - ii. Next to **Auto Select**, choose 'Yes'.
 - d. Click and drag the drop down below your 'Select Term:' label.
39. Write a SQL query to return a list of terms for the drop down menu.

- a. Double click on the Select Term drop down to open the **Choice Entry** window:
 - i. Select the **SQL Statement** radio button.
 - ii. Click the **Next** button.
 - iii. On the next screen, click on the **Edit Visual Design** button, to open the **Build Query** window.
 - (1) Click on the **Add Table** button at the top.
 - (a) Type 'STVTERM' and click the **OK** button.
 - (b) Select the 'STVTERM' table from the **Choose Table** window.
 - (c) Click the **OK** button.
 - (2) Click the **Visible Fields (SELECT)** tab.
 - (a) Double-click 'STVTERM_CODE' in the table window to add it to the **Visible Fields (SELECT)** tab.
 - (3) For this query, you're also going to create a second field in the SELECT statement that displays both the term code and the term description by creating a calculated field. On the **Visible Fields (SELECT)** tab:
 - (a) Click in the **Table** row (the top row) of the **second column**, and choose '<calculated>' from the drop down.
 - (b) Click in the **Field** row of the **second column**, and click on the ellipsis button to open the **SQL Editor** window:
 - (i) Click the **Insert Field** button (the green neutron-looking icon on the toolbar),
 
 - to pull up the **Pick a Field** window:
 - 1) Click the **+** button to expand out 'STVTERM'.
 - 2) Click 'STVTERM_CODE'.
 - 3) Click the **OK** button.
 - (ii) Back in the **SQL Editor**, after 'STVTERM.STVTERM_CODE', type the following:


```
|| '-' ||
```

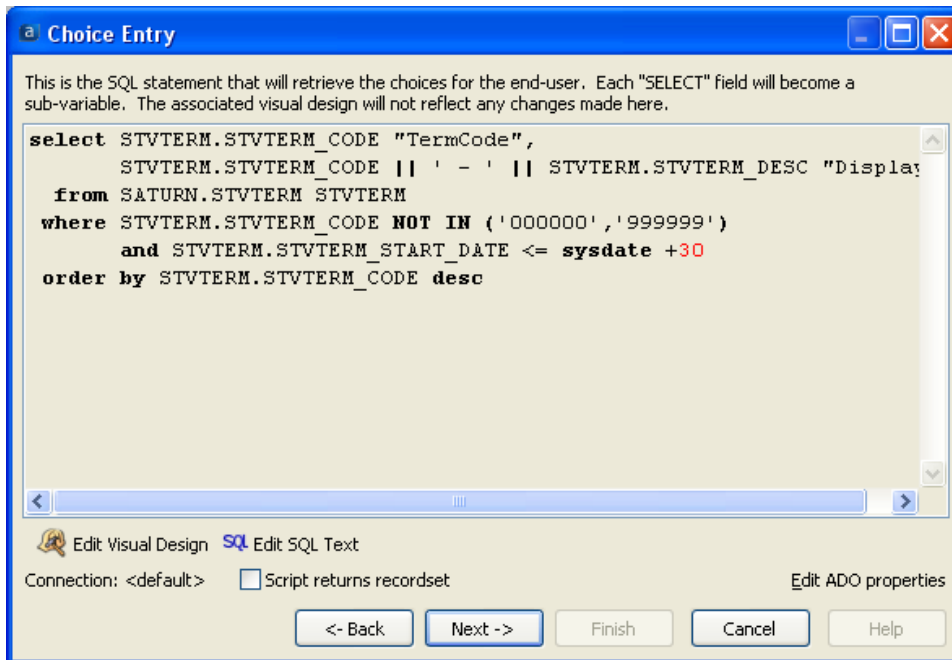
NOTE: The | character is usually on the key directly above your Enter key. This code will display a hyphen between the two values we're calculating.
 - (iii) Click the **Insert Field** button again, to add the second field to the calculation:
 - 1) Click the **+** button to expand out 'STVTERM'.
 - 2) Click 'STVTERM_DESC'.
 - 3) Click the **OK** button.
 - (iv) Confirm that your string is correct—it should read:


```
STVTERM.STVTERM_CODE || '-' || STVTERM.STVTERM_DESC
```
 - (v) Click the **OK** button.
 - (c) Back in the **Build Query** window, on the **Visible Fields (SELECT)** tab, click in the **As** row of the **second column**, and type 'Display' (instead of 'calc1').
 - (4) Click on the **Conditional Fields (WHERE)** tab, where you're going to add conditions to the query:
 - (a) Double-click 'STVTERM_CODE' in the table window to add it to the **Conditional Fields (WHERE)** tab.
 - (b) Click in the **Condition** row (the bottom row), and click on the ellipsis button to open the **SQL Editor** window:
 - (i) Type the following text:


```
not in ('000000', '999999')
```
 - (ii) Click the **OK** button.
 - (c) Double-click 'STVTERM_START_DATE' in the table window to add it to the **Conditional Fields (WHERE)** tab to include a second condition.
 - (d) Click in the **Condition** row, and click on the ellipsis button to open the **SQL Editor** window:
 - (i) Type the following text:

<= sysdate +30

- (ii) Click the **OK** button.
 - (5) Click on the **Ordering (ORDER BY)** tab.
 - (a) Double-click 'STVTERM_CODE' in the table window to add it to the **Ordering (ORDER BY)** tab.
 - (b) Click in the **Sort** row (the bottom row), and select 'Descending' from the drop down.
 - (6) Click the **OK** button to exit the Visual Designer.
- iv. Review the SQL query you just built. It should look like this:



- v. Click the **NEXT** button.
 - vi. After reviewing your first 5 results, click the **NEXT** button again.
 - vii. Click on the drop down and choose 'Display' from the list. This will display the calculated field in the list, showing users the combined Term code and Term name.
 - viii. Click the **Finish** button.
- b. Commit and test your changes. (Follow steps 24a-d.)

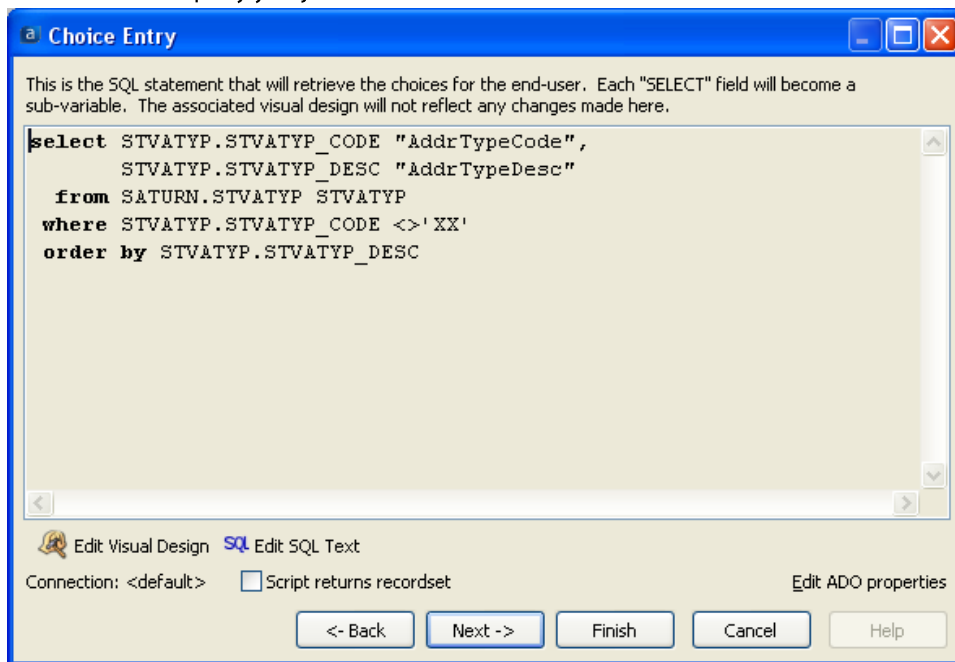
Add the 'Address Type' variable

40. Add a label for your variable.
 - a. Add a new label to the panel. (Follow steps 29-30.)
 - b. Double-click and change the text to 'Select Address Type:'.
 - c. Click and drag the label to the right side of the panel, under the Select Term drop down.
41. Add a drop down control object.
 - a. Click the **Add Drop Down** tool.
 - b. Click on the panel to place the drop down.
 - c. On the **Properties** tab:
 - i. Next to **Variable Name**, type 'main_DD_AddrType'.
 - d. Click and drag the drop down below your 'Select Address Type:' label.
42. Write a SQL query to return a list of address types for the drop down menu.
 - a. Double click on the Address Type drop down to open the **Choice Entry** window:

- i. Select the **SQL Statement** radio button.
- ii. Click the **Next** button.
- iii. On the next screen, click on the **Edit Visual Design** button, to open the **Build Query** window.
 - (1) Click on the **Add Table** button at the top.
 - (a) Type 'STVATYP' and click the **OK** button.
 - (b) Select the 'STVATYP' table from the **Choose Table** window.
 - (c) Click the **OK** button.
 - (2) Click the **Visible Fields (SELECT)** tab.
 - (a) Double-click 'STVATYP_CODE' in the table window to add it to the **Visible Fields (SELECT)** tab.
 - (b) Double-click 'STVATYP_DESC' in the table window to add it to the **Visible Fields (SELECT)** tab.
 - (3) Click the **Conditional Fields (WHERE)** tab.
 - (a) Double-click 'STVATYP_CODE' in the table window to add it to the **Conditional Fields (WHERE)** tab.
 - (b) Click in the **Condition** row, and type the following text:


```
<>'XX'
```

NOTE: You could also click the ellipsis button and add this text string in the SQL Editor, but because it's a very short string, this saves some time.
 - (4) Click the **Ordering (ORDER BY)** tab.
 - (a) Double-click 'STVATYP_DESC' to add it to the **Ordering (ORDER BY)** tab, and make sure 'Ascending' is selected in the **Sort** row.
 - (5) Click the **OK** button to exit the Visual Designer.
- iv. Review the SQL query you just built. It should look like this:



- v. Click the **Next** button.
 - vi. You should see a list of Address Type codes and descriptions. Click the **Next** button again.
 - vii. Click the drop down and choose 'AddrTypeDesc' to display the address type descriptions to the user.
 - viii. Click the **Finish** button.
- b. Commit and test your changes. (Follow steps 24a-d.)

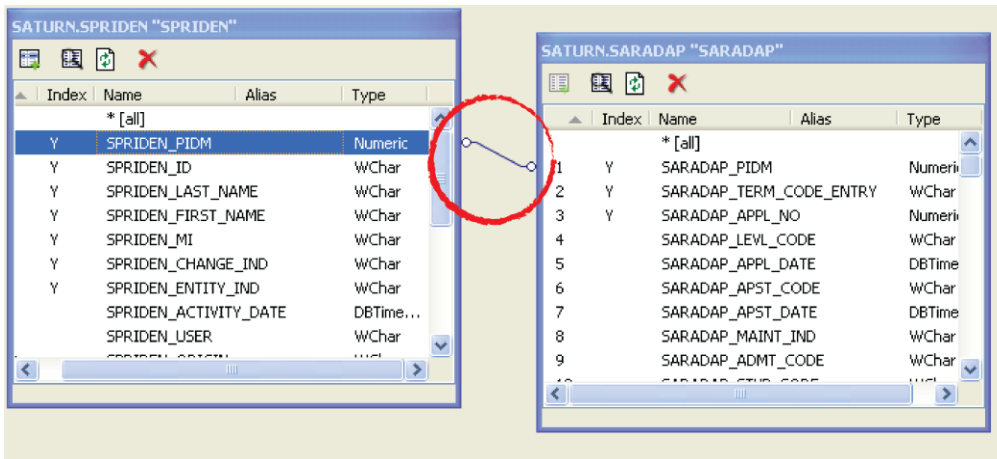
Create a Report Query for this DataBlock

43. In the upper left corner, click the **Report Query – Visual Design** tab:

- a. Click on the **Add Table** button at the top, and repeat the following three steps for these four tables: 'SPRIDEN', 'SPRADDR', 'SARADAP', 'STVADMT'
 - i. Type the table name, and click the **OK** button.
 - ii. Select that table from the **Choose Table** window.
 - iii. Click the **OK** button.

NOTE: Once you have all four table windows open in the Report Query - Visual Design tab, it's useful to rearrange them so you can see all four at once.

44. In order to create a reference link between the data in the tables, you're going to create joins between several of them. Let's start with 'SPRIDEN' and 'SARADAP':
 - a. In the 'SPRIDEN' table window, find 'SPRIDEN_PIDM' (row 1).
 - b. Click and drag 'SPRIDEN_PIDM' and drop it on 'SARADAP_PIDM' (also row 1) of the 'SARADAP' table window. There should now be a blue line connecting 'SPRIDEN_PIDM' to 'SARADAP_PIDM'.



45. Next, you're going to join 'SARADAP' to 'STVADMT':
 - a. In the 'SARADAP' table window, find 'SARADAP_ADMT_CODE,' which should be in row 9.
 - b. Click and drag 'SARADAP_ADMT_CODE' to 'STVADMT_CODE' in row 1 of the 'STVADMT' table window. There should now also be a blue line connecting 'SARADAP_ADMT_CODE' to 'STVADMT_CODE'.
46. To check your joins, click on the **View SQL** button (magnifying glass button on the toolbar).

47. Review your SQL. It should look like this:

```

from SATURN.SPRIDEN SPRIDEN,
SATURN.SPRADDR SPRADDR,
SATURN.SARADAP SARADAP,
SATURN.STVADMT STVADMT
where ( SPRIDEN.SPRIDEN_PIDM = SARADAP.SARADAP_PIDM
and SARADAP.SARADAP_ADMT_CODE = STVADMT.STVADMT_CODE )
and ( ( spriden_change_ind is null )
and ( spraddr.rowid (+) = f_get_address_rowid(SPRIDEN_PIDM, 'STDNADDR', 'A', SYSDATE, 1, 'S', NULL) ) )
  
```

48. Click the **Close** button.

49. Now to add some detail to your Report Query. Click on the **Visible Fields (SELECT)** tab.

a. Double-click on every field listed below to add them to the **Visible Fields (SELECT)** tab:

SPRIDEN	SPRADDR	SARADAP	STVADMT
SPRIDEN_ID	SPRADDR_ATYP_CODE	SARADAP_TERM_CODE_ENTRY	STVADMT_DESC
SPRIDEN_LAST_NAME	SPRADDR_STREET_LINE_1	SARADAP_LEVEL_CODE	
SPRIDEN_FIRST_NAME	SPRADDR_STREET_LINE_2	SARADAP_STYP_CODE	
	SPRADDR_CITY		
	SPRADDR_STAT_CODE		
	SPRADDR_ZIP		

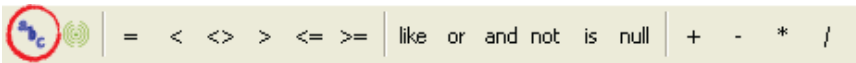
NOTE: Scroll to the right to see all of the fields you've added to the Visible Fields (SELECT) tab.

50. Next, you're going to add a few arguments to your query. Click on the **Conditional Fields (WHERE)** tab:

a. Double-click 'SARADAP_ADMT_CODE' in the 'SARADAP' table window to add it to the **Conditional Fields (WHERE)** tab.

b. Click in the **Condition** row, and click on the ellipsis button to open the **SQL Editor** window:

i. Click on the **User-Defined Variable** button (the ABC button on the toolbar)



to bring up a list of DataBlock variables:

- (1) Click the **+** button to expand out the 'main_LB_AcceptCode' list.
- (2) Select 'AdmissTypeCode'.
- (3) Click the **OK** button.

ii. Back in the **SQL Editor** click the **OK** button again.

NOTE: When you are adding a variable to the Condition row, be aware that if do not supply an operator, Argos will add an equal operator (=) automatically. The full condition you're building here is 'SARADAP_ADMT_CODE

=:main_LB_AcceptCode'. In the Condition row, you can choose to type the '=' before specifying the variable or just let Argos supply the '=' for you.

51. Test your new condition:
 - a. Click the **Commit** button to save.
 - b. Click the **Test** button to preview the report query.
 - c. Select an Admission Type from the **Admission Acceptance Type** list box.
 - d. There is now a pane at the bottom of the preview window where you can run a test report:
 - i. Enter the number of records you would like to see in the **Retrieve a maximum records** text box in the report query test pane.
 - ii. Click the **Get** button to make sure your results match your variable selection.
 - iii. Click the **Close** button to exit the test window.
52. To create the next condition, click on the **Conditional Fields (WHERE)** tab:
 - a. Double-click on 'SARADAP_TERM_CODE_ENTRY' in the 'SARADAP' table window to add it to the **Conditional Fields (WHERE)** tab.
 - b. Click in the **Condition** row, and click on the ellipsis button to open the **SQL Editor** window:
 - i. Click on the **User-Defined Variable** button to bring up a list of DataBlock variables.
 - (1) Click the + button to expand out the 'main_DD_Term' list.
 - (2) Select 'TermCode'.
 - (3) Click the **OK** button.
 - ii. Back in the **SQL Editor** click the **OK** button again.
53. Test your new condition:
 - a. Click the **Commit** button to save.
 - b. Click the **Test** button to preview the report query.
 - c. Select an Admission Type from the **Admission Acceptance Type** list box.
 - d. Select a term from the **Select Term** drop down.
 - e. In the test report pane:
 - i. Click the **Get** button to see your results.
 - ii. Click the **Close** button to exit the test window.
54. To create the last condition for this report query, click the **Conditional Fields (WHERE)** tab.
 - a. Double-click on 'SPRADDR_ATYP_CODE' in the 'SPRADDR' table window to add it to the **Conditional Fields (WHERE)** tab.
 - b. Click in the **Condition** row, and click on the ellipsis button to open the **SQL Editor** window:
 - i. Click on the **User-Defined Variable** button to bring up a list of DataBlock variables.
 - (1) Click the + button to expand out the 'main_DD_AddrType' list.
 - (2) Select 'AddrType_Code'.
 - (3) Click the **OK** button.
 - ii. Back in the **SQL Editor** click the **OK** button again.
55. Test your new condition:
 - a. Click the **Commit** button to save.
 - b. Click the **Test** button to preview the report query.
 - c. Select an Admission Type from the **Admission Acceptance Type** list box.
 - d. Select a term from the **Select Term** drop down.
 - e. Select an address type from the **Address Type** drop down.
 - f. In the test report pane:
 - i. Click the **Get** button to see your results.
 - ii. Click the **Close** button to exit the test window.

56. One last thing—you're going to add a sort order. Click **Ordering (ORDER BY)** tab:
 - a. Double-click on 'SPRIDEN_ID' in the 'SPRIDEN' table window to add it to the **Ordering (ORDER BY)** tab, and make sure the **Sort** row is set to 'Ascending'.
57. Click on the **View SQL** button (magnifying glass button on the toolbar) to review your complete Report Query. It should look like this:

```

select SPRIDEN.SPRIDEN_ID "Id",
       SPRIDEN.SPRIDEN_LAST_NAME "LastName",
       SPRIDEN.SPRIDEN_FIRST_NAME "FirstName",
       SPRADDR.SPRADDR_ATYP_CODE "AddrType",
       SPRADDR.SPRADDR_STREET_LINE1 "StreetLine1",
       SPRADDR.SPRADDR_STREET_LINE2 "StreetLine2",
       SPRADDR.SPRADDR_CITY "City",
       SPRADDR.SPRADDR_STAT_CODE "State",
       SPRADDR.SPRADDR_ZIP "ZipCode",
       SARADAP.SARADAP_TERM_CODE_ENTRY "TermCodeEntry",
       SARADAP.SARADAP_LEVL_CODE "LevelAppCode",
       SARADAP.SARADAP_STYP_CODE "StudTypeCode",
       STVADMT.STVADMT_DESC "AdmissType"
from SATURN.SPRIDEN SPRIDEN,
     SATURN.SPRADDR SPRADDR,
     SATURN.SARADAP SARADAP,
     SATURN.STVADMT STVADMT
where ( SPRIDEN.SPRIDEN_PIDM = SARADAP.SARADAP_PIDM
       and SARADAP.SARADAP_ADMT_CODE = STVADMT.STVADMT_CODE )
       and ( ( SARADAP.SARADAP_ADMT_CODE = :main_LB_AcceptCode.AdmissTypeCode
             and SARADAP.SARADAP_TERM_CODE_ENTRY = :main_DD_Term.TermCode
             and SPRADDR.SPRADDR_ATYP_CODE = :main_DD_AdrType.AddrTypeCode )
         and ( spriden_change_ind is null )
         and ( spraddr.rowid (+) = f_get_address_rowid(SPRIDEN_PIDM, 'STDNADDR', 'A', SYSDATE, 1, 'S', NULL) ) )
order by SPRIDEN.SPRIDEN_ID
  
```

58. Test your complete Report Query:
 - a. Click the **Commit** button to save.
 - b. Click the **Test** button to preview the report query.
 - c. Select an Admission Type from the **Admission Acceptance Type** list box.
 - d. Select a term from the **Select Term** drop down.
 - e. Select an address type from the **Address Type** drop down.
 - f. In the test report pane:
 - i. Click the **Get** button to see your results.
 NOTE: They should be the same as last time, only sorted by the first column.
 - g. Click the **Close** button to exit the test window.
59. If you have not already done so, click the **Commit** button to save your changes.
60. Click the **Close** button to exit the DataBlock Designer. You've created your first DataBlock!

Supplemental Exercise: Create a DataBlock Template

Using the DataBlock you created in Exercise 1, 'Address List for Admitted Students,' you're going to create a DataBlock template to make it easy to create many DataBlocks with the same look.

Make a copy of the DataBlock

61. In the **Explorer** tab on the left side of the screen, click on your 'Address List for Admitted Students' DataBlock to highlight it.
62. Right-click on the DataBlock, and select **Copy** from the popup menu.
63. Click on the folder where you want to put the copy.
NOTE: Saving it in your user folder with the original DataBlock is fine.
64. Right-click on the folder, and select **Paste** from the popup menu.
65. Click the **Paste** button at the bottom of the **Paste** window. You should now have a new DataBlock called 'Address List for Admitted Students (2)'.

Edit the DataBlock to remove objects you don't want in the template

66. Click to highlight the duplicate DataBlock, 'Address List for Admitted Students (2)'.
67. On the right side of the screen, under the **DataBlock Designer Actions** heading, click the **Edit** button.
68. Double-click on the 'Find addresses for admitted students' label to open the **Modify Text** window:
 - a. Replace the text with 'Description/Instructions'.
 - b. Click the **OK** button.
69. Back in the main window, you're going to delete the controls specific to the 'Address List for Admitted Students' DataBlock. This way, when you use this template, you'll start with a new DataBlock that has a Title, Description, your school's logo, and nothing else. To delete all of the elements on your form:
 - a. Click an empty spot on your form to deselect the 'Description/Instructions' label.
 - b. Hold down the **Shift** key.
 - c. While holding the **Shift** key, click on the six objects on your form (three controls and their corresponding labels) to select them.
NOTE: Be careful not to click on the panel object.
 - d. Release the **Shift** key.
 - e. Click on the **Delete** button (the red X on the toolbar) to delete the objects.



NOTE: You can also delete the selected objects by hitting the Delete key on your keyboard.

Add remaining objects to the Library of Objects to create a template

70. Hold down the **Shift** key.
71. Click on the white panel object to select it.
NOTE: You need to select the panel object first—otherwise it will deselect anything else you have selected when you click on it.
72. While still holding the **Shift** key, click on the other objects on your form: the '\$DataBlock.Name' label and your school's logo.
73. Release the **Shift** key.

74. Click the **Add to Library of Objects** button (the stack of books with the green + on the toolbar)



to open the **Add Object** window:

- a. Your user folder should be automatically selected. (If it's not, click on it to select it.)
- b. In the **Name** text box, type in 'Training DataBlock Template.' This will save this group of form objects to the Library of Objects.
- c. Click the **OK** button.
- d. To verify the form is in the Library of Objects click the Add Objects from Library button (the stack of books(with no +) on the toolbar)



- e. In the Browse box you will see your userfolder name and underneath your new template.
- f. Click on cancel to close the Browse box.

75. Back in the DataBlock editor, click the **Close** button to exit, and click the **No** button when it asks if you want to commit your changes.

NOTE: You do not need to save your changes to this DataBlock, since you have just saved them in the Library of Objects.

76. Under the **DataBlock Designer Actions** heading, click the **Delete** button to delete the duplicate DataBlock, as it is no longer needed.

Second Exercise: Basic DataBlock 2 – Budget Availability

Basic Datablock 2 - Budget Availability

argos[®]
powered by evisions

Chart of Accounts: A - SCT University Fiscal Year: 94

Fund: Orgn: Account:

Run Dashboard

Fund	Orgn	Acct	Prog	TotalAdoptBudg	TotalBudgAdjust	TotalYTDActiv	TotalEncumb	TotalBudgReserv	FiskaPeriod
0 items									

Products Training Support Feedback www.evisions.com

Exercise Description

This DataBlock includes a multi-column list box that—based on user selections—will display a set of budget data. The report query is based on the multi-column list box's SQL query and will allow users to run reports with that same data set. Above is an example of what the finished DataBlock will look like at the end of the exercise.

In addition to using your DataBlock template to start the layout for this form, you'll also learn how to add cascading variables, i.e. the selections available in each control box will depend on what the user selected in the previous boxes. Here is a description of each of the five cascading variables you'll be adding to this DataBlock along with a note on which database table they draw from:

Chart of Accounts – drop down box that returns the chart of accounts users can choose from. *Table: FTVCOAS*

Fiscal Year – drop down box that returns a list of fiscal years from which data is available for the chosen chart of accounts. *Table: FTVFSYR*

Fund – list box that returns a list of funds available based on the chosen chart of accounts and fiscal year. *Table: FGGBAVL*

Organization – list box that returns a list of organization codes based on the chosen chart of accounts, fiscal year, and funds. *Table: FGGBAVL*

Account – list box that returns a list of account codes based on the chosen chart of accounts, fiscal year, funds, and organizations. *Table: FGGBAVL*

You'll also be adding two other elements to the DataBlock, which will add functionality for the end user:

Run Dashboard button – button that allows the end user to complete all their variable selections and then run a query that returns a data set, which will be displayed in the multi-column list box. *Table: FGGBAVL*

Multi-Column List Box – when the Run Dashboard button is clicked, it returns the requested data set, based on the 5 cascading variables, and displays them in this list box. *Table: FGGBAVL*

And as with the previous DataBlock, you'll also be building a Report Query to enable end users to run reports off the DataBlock:

'Budget Availability' Report Query – allows users to create reports with the data in this DataBlock. *Tables: FTVCOAS, FGGBAVL, FTVFUND, FTVORGN, FTVACCT*

Instructions

NOTE: The instructions in this and subsequent exercises will be slightly less detailed than the first exercise. Please refer to earlier sections or to the Argos in-product help if you need additional information about specific steps.

Create a new DataBlock and open it for editing

77. Find your user folder in the folder list on the left-hand side.
78. Right-click on your folder, and choose **New > DataBlock** from the menu.
79. Type in the name of your new DataBlock – 'Budget Availability' – and hit the **Enter** key.
80. Click on your new DataBlock to highlight it.
81. On the right side of the screen, under the **DataBlock Designer Actions** heading, select the ADO being used for your training from the Associated Connection/Pool drop down.
82. Click the **Edit** button to open the DataBlock Designer.

Configure the form layout and add template objects

83. Click on the **Properties** tab:
 - a. Next to **Color**, select your background color from the drop down.
84. Click on the **Add Objects from Library** button (the stack of books (with no +) on the toolbar)



to open the **Choose an Object** window:

- a. Your user folder should be automatically selected. (If it's not, click on it to select it.)
 - b. Click on 'Training DataBlock Template' to select it.
 - c. Click the **OK** button.
85. Back in the **DataBlock Designer** window, you'll see that all your template objects have been added to the form.
 86. Click on the 'Description/Instructions' label object to select it.
 87. Click the **Delete** button (or the **Delete** key on your keyboard) to remove the label from the form.

Add the 'Chart of Accounts' variable

88. Add a label for your variable.
 - a. Add a new label to the panel. *(Follow steps 29-30.)*
 - b. Double-click and change the text to 'Chart of Accounts'.
 - c. On the **Properties** tab:
 - i. Next to **Font.Bold**, choose 'Yes'.
 - d. Click and drag the label to the upper-left corner of the panel.
89. Add a drop down control object.
 - a. Click the **Add Drop Down** tool.
 - b. Click on the panel to place the drop down.

- c. On the **Properties** tab:
 - i. Next to **Variable Name**, type 'main_DD_COA'.
 - ii. Next to **Auto Select**, choose 'Yes'.
 - d. Click and drag the drop down next to your 'Chart of Accounts:' label.
90. Write a SQL query to return a chart of accounts list for the drop down menu.
- a. Double click on the 'Chart of Accounts' drop down to open the **Choice Entry** window:
 - i. Select the **SQL Statement** radio button.
 - ii. Click the **Next** button.
 - iii. On the next screen, click on the **Edit Visual Design** button, to open the **Build Query** window.
 - (1) Click on the **Add Table** button at the top.
 - (a) Type 'FTVCOAS' and click the **OK** button.
 - (b) Select the 'FTVCOAS' table from the **Choose Table** window.
 - (c) Click the **OK** button.
 - (2) Click the **Visible Fields (SELECT)** tab.
 - (a) Double-click 'FTVCOAS_COAS_CODE' in the table window to add it to the **Visible Fields (SELECT)** tab.
 - (b) Click on the **Summing** button to display the summing row.
 - (c) Click in the **Summing** row and select '<Group By>' from the drop down.
 - (d) You're also going to create a calculated field to display the code and description of the chart of accounts. Click in the **Table** row (the top row) of the second column, and choose '<calculated>' from the drop down.
 - (e) Click in the **Field** row of the second column, and click on the ellipsis button to open the **SQL Editor** window:
 - (i) Click the **Insert Field** button (the green neutron-looking icon on the toolbar) to pull up the **Pick a Field** window:
 - 1) Click the + button to expand out 'FTVCOAS'.
 - 2) Click 'FTVCOAS_COAS_CODE'.
 - 3) Click the **OK** button.
 - (ii) Back in the **SQL Editor**, after 'FTVCOAS.FTVCOAS_COAS_CODE', type the following:
 || ' - ' ||
 - (iii) Click the **Insert Field** button again, to add the second field to the calculation:
 - 1) Click the + button to expand out 'FTVCOAS'.
 - 2) Click 'FTVCOAS_TITLE'.
 - 3) Click the **OK** button.
 - (iv) Confirm that your string is correct—it should read:
 FTVCOAS.FTVCOAS_COAS_CODE || ' - ' || FTVCOAS.FTVCOAS_TITLE
 - (v) Click the **OK** button.
 - (f) Click in the **As** row and type 'Title'.
 - (3) Click the **Ordering (ORDER BY)** tab.
 - (a) Double-click on 'FTVCOAS_COAS_CODE' to add it to the **Ordering (ORDER BY)** tab.
 - (4) Click the **OK** button to exit the Visual Designer.

NOTE: From now on, when you see 'Open the Visual Designer,' 'Add Table: [NAME OF TABLE],' or 'Add [NAME OF FIELD] to Visible Fields (SELECT) or Ordering (ORDER BY) tab' follow the steps in the preceding section.

 - iv. Review the SQL query you just built. It should look like this:

```

select FIVCOAS.FIVCOAS_COAS_CODE "ChartAcctCode",
       FIVCOAS.FIVCOAS_COAS_CODE || ' - ' || FIVCOAS.FIVCOAS_TITLE "Title"
from FIVCOAS.FIVCOAS
group by FIVCOAS.FIVCOAS_COAS_CODE,
         FIVCOAS.FIVCOAS_COAS_CODE || ' - ' || FIVCOAS.FIVCOAS_TITLE
order by FIVCOAS.FIVCOAS_COAS_CODE

```

- v. Click the **Next** button.
- vi. You should see a list of the Chart of Accounts. Click the **Next** button.
- vii. Select 'Title' from the drop down to use that as the field displayed to the users.
- viii. Click the **Finish** button.

91. Commit and test your changes. (Follow steps 24a-d.)

Add the 'Fiscal Year' variable

- 92. Add a label for your variable.
 - a. Add a new label to the panel. (Follow steps 29-30.)
 - b. Double-click and change the text to 'Fiscal Year:'.
 - c. On the **Properties** tab:
 - i. Next to **Font.Bold**, choose 'Yes'.
 - d. Click and drag the label to the right of the 'Chart of Accounts' drop down.

93. Add a drop down control object.

- a. Click the **Add Drop Down** tool.
- b. Click on the panel to place the drop down.
- c. On the **Properties** tab:
 - i. Next to **Variable Name**, type 'main_DD_FiscalYr'.
 - ii. Next to **Auto Select**, choose 'Yes'.
- d. Click and drag the drop down next to your 'Fiscal Year:' label.

94. Write a SQL query to return a list of fiscal years for the drop down menu.

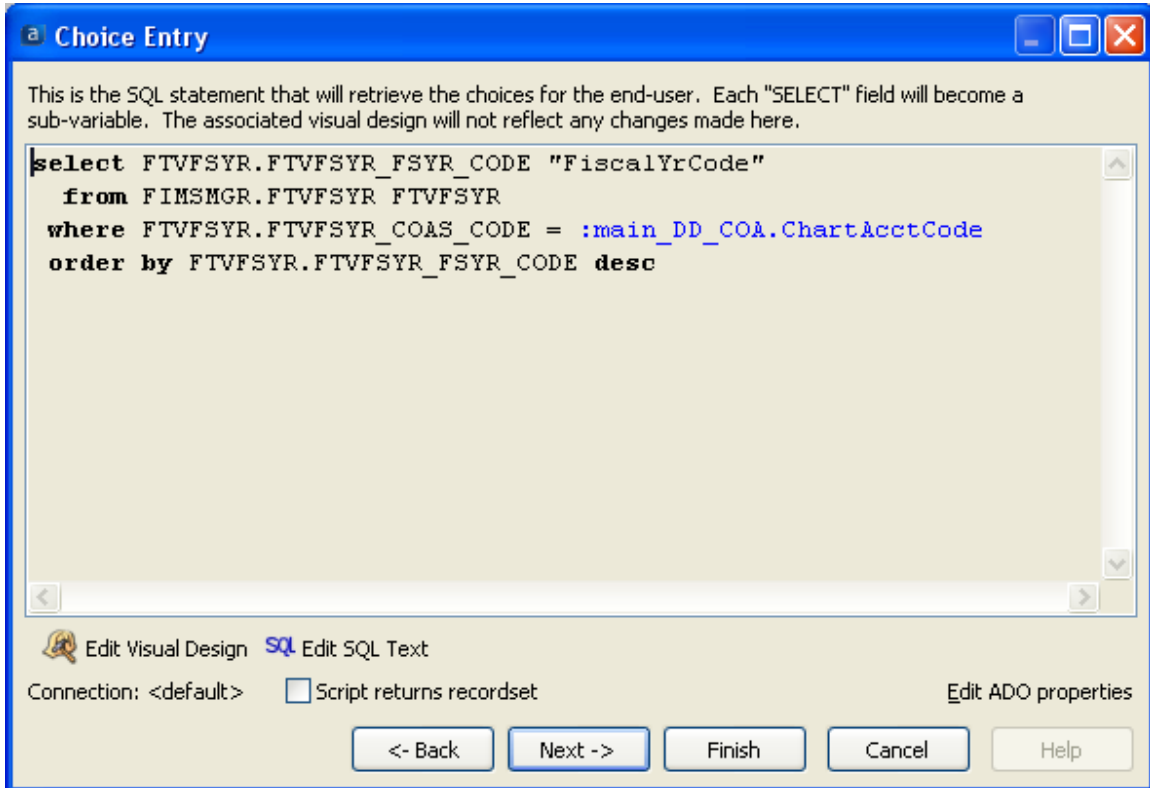
- a. Double click on the 'Fiscal Year' drop down.
 - i. Open the Visual Designer.

Remember: For more details on working in the Visual Designer, see instructions in step 90.

 - (1) Add Table: FTVFSYR
 - (2) Add 'FTVFSYR_FSYR_CODE' to the **Visible Fields (SELECT)** tab.
 - (3) Click on the **Conditional Fields (WHERE)** tab:
 - (a) Double-click 'FTVFSYR_COAS_CODE' in the 'FTVFSYR' table window to add it to the **Conditional Fields (WHERE)** tab.
 - (b) Click in the **Condition** row, and click on the ellipsis button to open the **SQL Editor** window:
 - (i) Click on the **User-Defined Variable** button (the ABC button on the toolbar) to bring up a list of DataBlock variables:
 - 1) Click the + button to expand out the 'main_DD_COA' list.
 - 2) Select 'ChartAcctCode'.
 - 3) Click the **OK** button.
 - (ii) Back in the **SQL Editor**, check that it says ':main_DD_COA.ChartAcctCode'.
 - (iii) Click the **OK** button again.

NOTE: From now on, when you see 'Add [NAME OF FIELD] to the Conditional Fields (WHERE) tab and add [NAME OF VARIABLE] to the Condition row', follow the steps in the preceding section.

- (4) Add 'FTVFSYR_FSYR_CODE' to the **Ordering (ORDER BY)** tab.
 - (a) Click in the **Sort** row and select 'Descending'.
 - (5) Click the **OK** button to exit the Visual Designer.
- ii. Review the SQL query you just built. It should look like this:



- iii. Click the **Next** button.
 - iv. A new window—called **Test Values**—will appear, asking you to supply test values for the cascading variables on the form. Leave everything blank and hit the **OK** button. (You'll do a test run when you've finished adding variables.)
 - v. A blank row should return. You won't see any values returned since you didn't enter any test values. Click the **Next** button.
 - vi. Confirm that 'FiscalYrCode' is selected as the field to be displayed to the users.
 - vii. Click the **Finish** button.
95. Commit and test your changes. (Follow steps 24a-d.)

Add the 'Fund' variable

96. Add a label for your variable.
 - a. Add a new label to the panel. (Follow steps 29-30.)
 - b. Double-click and change the text to 'Fund:'.
 - c. On the **Properties** tab:
 - i. Next to **Font.Bold**, choose 'Yes'.
 - d. Click and drag the label under the 'Chart of Accounts:' label.
97. Add a list box control object.
 - a. Click the **Add List Box** tool.

- b. Click on the panel to place the list box.
- c. On the **Properties** tab:
 - i. Next to **Variable Name**, type 'main_LB_Fund'.
 - ii. Next to **Multi Select**, choose 'Yes'.
- d. Click and drag the list box next to your 'Fund:' label.

98. Write a SQL query to return a list of funds for the list box.

- a. Double click on the 'Fund' list box.
 - i. Open the Visual Designer.

Remember: For more details on working in the Visual Designer, see instructions in steps 90 & 94.

(1) Add Table: FGGBAVL

(2) Add 'FGGBAVL_FUND_CODE' to the **Visible Fields (SELECT)** tab.

- (a) Click on the **Summing** button to display the summing row.
- (b) Click in the **Summing** row and select '<Group By>' from the drop down.
- (c) Click in the **As** row and type 'FundCode'.

NOTE: Although the data dictionary will often supply an alias for a given field, if there isn't one, you can always supply it yourself by typing it into the As row.

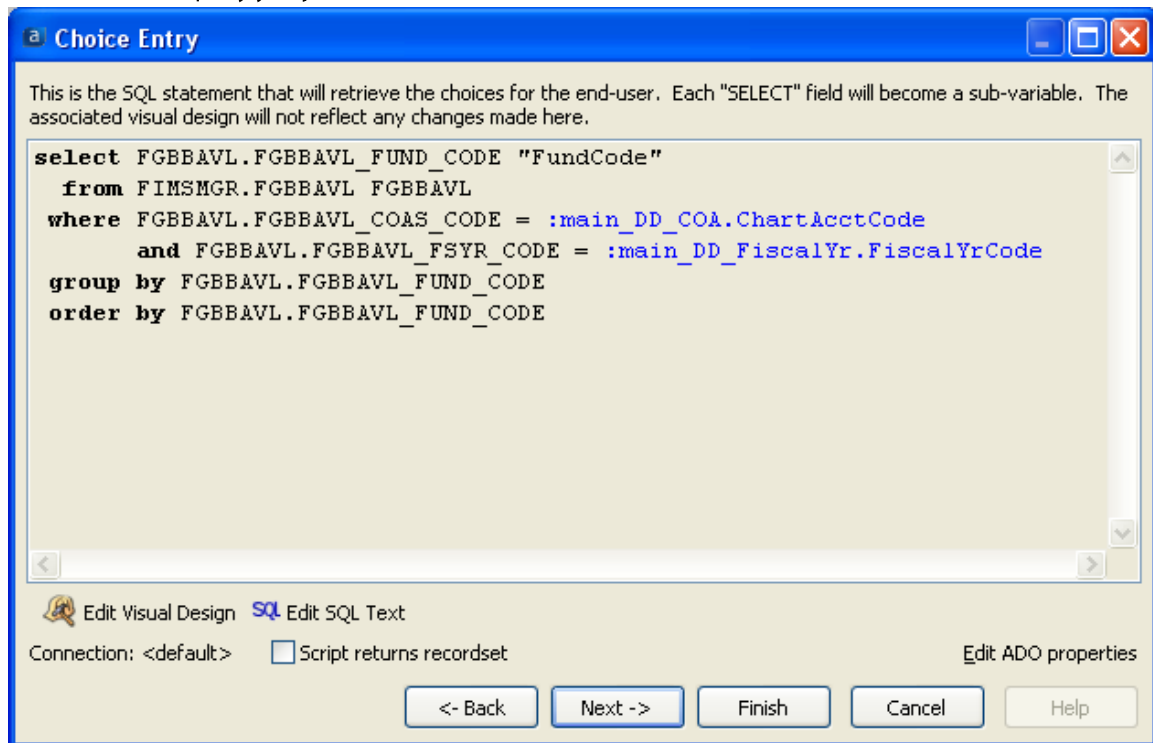
(3) Add the following fields to the **Conditional Fields (WHERE)** tab, and add the following variables to the **Condition** row:

Field	Condition
FGGBAVL_COAS_CODE	:main_DD_COA.ChartAcctCode
FGGBAVL_FSYR_CODE	:main_DD_FiscalYr.FiscalYrCode

(4) Add 'FGGBAVL_FUND_CODE' to the **Ordering (ORDER BY)** tab.

(5) Click the **OK** button to exit the Visual Designer.

- ii. Review the SQL query you just built. It should look like this:



- iii. Click the **Next** button.
- iv. Click the **OK** button on the **Test Values** window.

- v. A blank row should return. You won't see any values returned since you didn't enter any test values. Click the **Next** button.
- vi. Confirm that 'FundCode' is selected as the field to be displayed to the users.
- vii. Click the **Finish** button.

99. Commit and test your changes. (Follow steps 24a-d.)

Add the 'Organization' variable

100. Add a label for your variable.

- a. Add a new label to the panel. (Follow steps 29-30.)
- b. Double-click and change the text to 'Orgn:'.
- c. On the **Properties** tab:
 - i. Next to **Font.Bold**, choose 'Yes'.
- d. Click and drag the label to the right of the Fund list box.

101. Add a list box control object.

- a. Click the **Add List Box** tool.
- b. Click on the panel to place the list box.
- c. On the **Properties** tab:
 - i. Next to **Variable Name**, type 'main_LB_Orgn'.
 - ii. Next to **Multi Select**, choose 'Yes'.
- d. Click and drag the drop down next to your 'Orgn:' label.

102. Write a SQL query to return a list of organizations for the list box.

- a. Double-click on the 'Orgn' list box.
 - i. Open the Visual Designer.

Remember: For more details on working in the Visual Designer, see instructions in steps 90 & 94.

(1) Add Table: FGGBAVL

(2) Add 'FGGBAVL_ORGN_CODE' to the **Visible Fields (SELECT)** tab.

- (a) Click on the **Summing** button to display the summing row.
- (b) Click in the **Summing** row and select '<Group By>' from the drop down.
- (c) Click in the **As** row and change the alias to 'OrganizationCode'

NOTE: Other times, you may want to change the alias provided by the data dictionary, which you can do by simply editing the value in the As row.

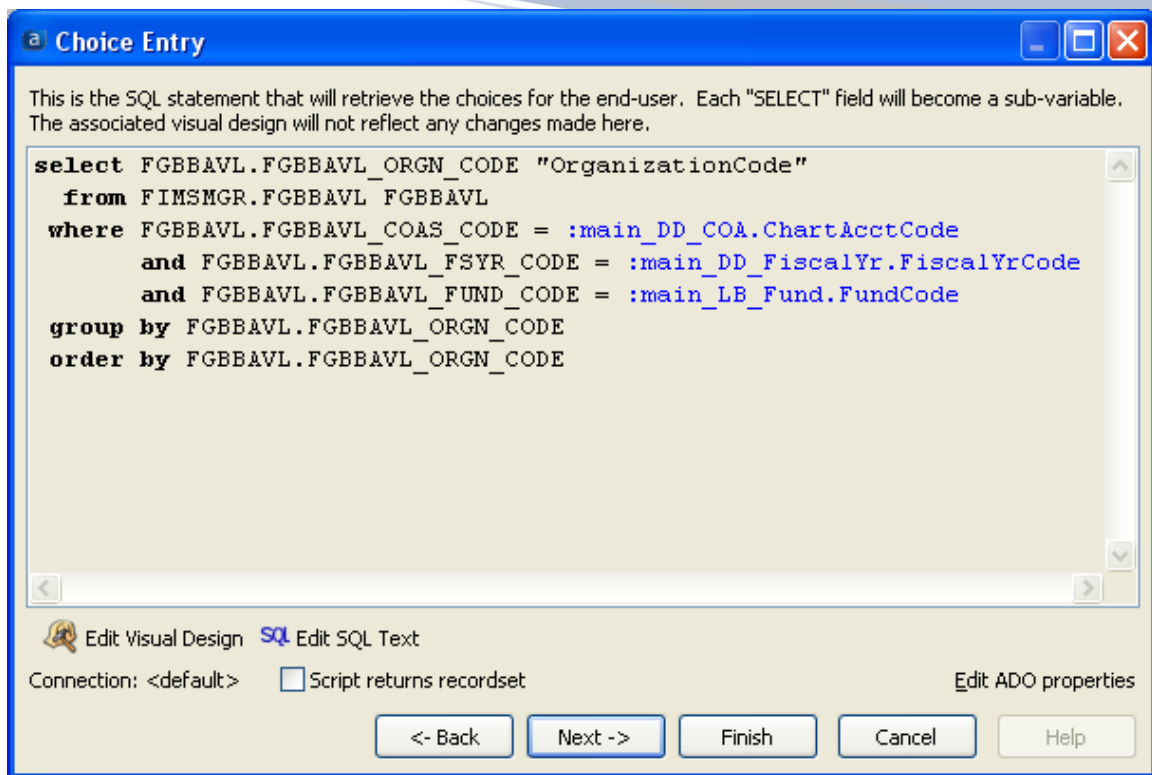
(3) Add the following fields to the **Conditional Fields (WHERE)** tab, and add the following variables to the **Condition** row:

Field	Condition
FGGBAVL_COAS_CODE	:main_DD_COA.ChartAcctCode
FGGBAVL_FSYR_CODE	:main_DD_FiscalYr.FiscalYrCode
FGGBAVL_FUND_CODE	:main_LB_Fund.FundCode

(4) Add 'FGGBAVL_ORGN_CODE' to the **Ordering (ORDER BY)** tab.

(5) Click the **OK** button to exit the Visual Designer.

- ii. Review the SQL query you just built. It should look like this:



- iii. Click the **Next** button.
- iv. Click the **OK** button on the **Test Values** window.
- v. A blank row should return. You won't see any values returned since you didn't enter any test values. Click the **Next** button.
- vi. Confirm that 'OrganizationCode' is selected as the field to be displayed to the users.
- vii. Click the **Finish** button.

103. Commit and test your changes. (Follow steps 24a-d.)

Add 'Account' variable

104. Add a label for your variable.
 - a. Add a new label to the panel. (Follow steps 29-30.)
 - b. Double-click and change the text to 'Account:'.
 - c. On the **Properties** tab:
 - i. Next to **Font.Bold**, choose 'Yes'.
 - d. Click and drag the label to the right of the 'Orgn' list box.
105. Add a list box control object.
 - a. Click the **Add List Box** tool.
 - b. Click on the panel to place the list box.
 - c. On the **Properties** tab:
 - i. Next to **Variable Name**, type 'main_LB_Acct'.
 - ii. Next to **Multi Select**, choose 'Yes'.
 - d. Click and drag the drop down next to your 'Account:' label.

106. Write a SQL query to return a list of accounts for the list box.

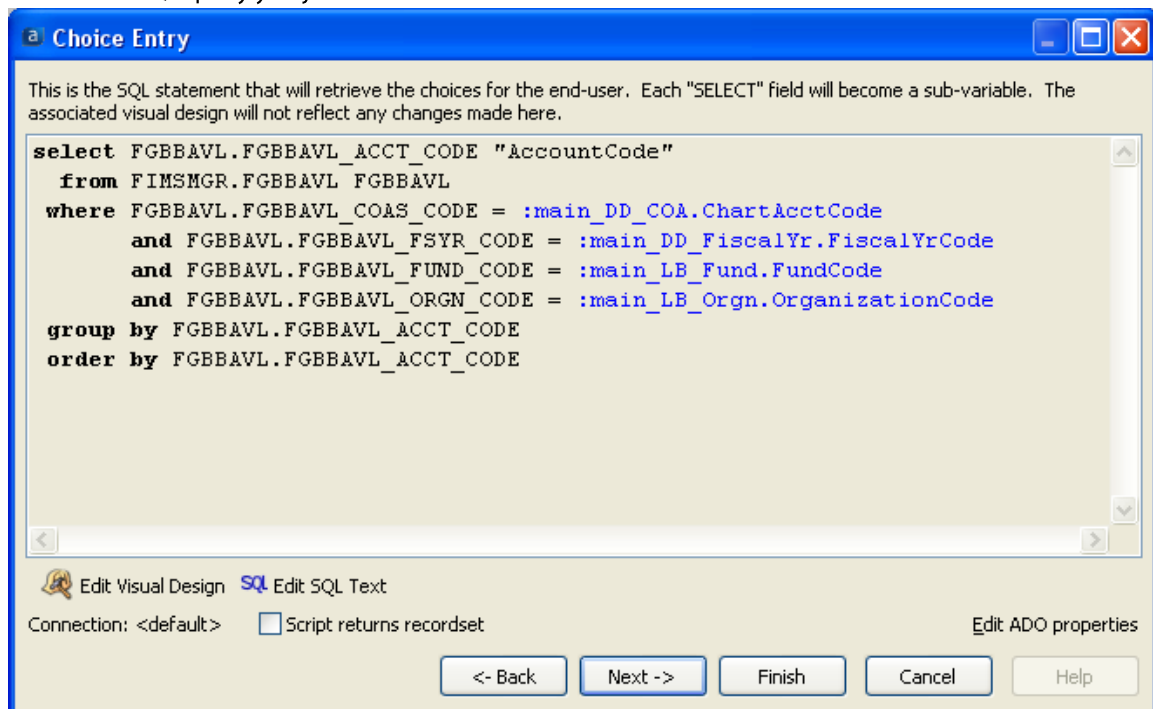
- a. Double-click on the 'Account' list box.
 - i. Open the Visual Designer.

Remember: For more details on working in the Visual Designer, see instructions in steps 90 & 94.

- (1) Add Table: FGGBAVL
- (2) Add 'FGGBAVL_ACCT_CODE' to the **Visible Fields (SELECT)** tab.
 - (a) Click on the **Summing** button to display the summing row.
 - (b) Click in the **Summing** row and select '<Group By>' from the drop down.
- (3) Add the following fields to the **Conditional Fields (WHERE)** tab, and add the following variables to the **Condition** row:

Field	Condition
FGGBAVL_COAS_CODE	:main_DD_COA.ChartAcctCode
FGGBAVL_FSYR_CODE	:main_DD_FiscalYr.FiscalYrCode
FGGBAVL_FUND_CODE	:main_LB_Fund.FundCode
FGGBAVL_ORGN_CODE	:main_LB_Orgn.OrganizationCode

- (4) Add 'FGGBAVL_ACCT_CODE' to the **Ordering (ORDER BY)** tab.
 - (5) Click the **OK** button to exit the Visual Designer.
- ii. Review the SQL query you just built. It should look like this:



- iii. Click the **Next** button.
- iv. Click the **OK** button on the **Test Values** window.
- v. A blank row should return. You won't see any values returned since you didn't enter any test values. Click the **Next** button.
- vi. Confirm that 'AccountCode' is selected as the field to be displayed to the users.
- vii. Click the **Finish** button.

107. Commit and test your changes. (Follow steps 24a-d.)

Add the 'Run Dashboard' button

108. Click the **Add Button** tool (the OK button in the toolbar).



109. Click on the panel to add the button.

110. On the **Properties** tab:

- Next to **Variable Name**, type 'main_BT_RunDashboard'.
- Next to **Font.Bold**, choose 'Yes'.
- Next to **Tab Stop**, choose 'Yes'

NOTE: This option will make it possible for the end user to cycle to the Run Dashboard button using the tab key. Many other types of control objects have this option selected by default.

- Next to **Text**, type 'Run Dashboard'

NOTE: Unlike a label, you cannot double-click on a button to edit the text.

111. Click and drag one of the button's corners to resize to fit the text.

Align the position of your objects

112. Refer to the picture of the finished Budget Availability DataBlock located at the beginning of this exercise, and click and drag all the objects to approximately the same place.

113. Click to select the 'Chart of Accounts' (COA) label. Hold the **Shift** key, and then click to select the COA drop down, the 'Fiscal Year' label, and the Fiscal Year drop down.

114. Release the **Shift** key and click the **Align by Vertical Centers** button (three blocks with an up arrow and a down arrow on the toolbar).



115. Hold the **Shift** key and click to select the labels and list boxes for 'Fund', 'Organization', and 'Account' (six objects total).

116. Release the **Shift** key and click the **Align by Top Sides** button (two blocks with an up arrow on the toolbar).



117. Hold the **Shift** key one more time and click to select 'Account' list box and the 'Run Dashboard' button.

118. Release the **Shift** key and click the **Align by Bottom Sides** button (two blocks with a down arrow on the toolbar).



NOTE: For more information about how to align objects on a form, please refer to the DataBlocks > Form Design – Visual Designer section of the Argos in-product help.

Create the multi-column list box to display budget information

119. Add a multi-column list box control object.

- Click on the **Add Multi-Column List** Box tool (the button on the toolbar that shows a list with two columns).



- Click in the panel to place the multi-column list box.
- Click and drag one of multi-column list box's corners to increase its size until it fills the available space on the panel below the list boxes above.

NOTE: When a user hits the Run Dashboard button on this form, their data set will be displayed in this control box. You want to make the multi-column list box as large as possible so it can display a lot of data without the end user having to scroll back and forth.

- On the **Properties** tab:
 - Next to **Show Item Count**, choose 'Yes'.
 - Next to **Variable Name**, type 'main_MC_AccountBalances'.

120. Write a SQL query to return the budget data set specified by the cascading variables.

a. Double-click on the multi-column list box.

i. Open the Visual Designer.

Remember: For more details on working in the Visual Designer, see instructions in steps 90 & 94.

(1) Add Table: FGGBAVL

(2) Add all of the fields below to the **Visible Fields (SELECT)** tab.

Field <i>use in step (2), above</i>	Alias <i>use in step (2a), below</i>
FGGBAVL_FUND_CODE	Fund
FGGBAVL_ORGN_CODE	Orgn
FGGBAVL_ACCT_CODE	Acct
FGGBAVL_PROG_CODE	Prog
FGGBAVL_SUM_ADOPT_BUD	TotalAdoptedBudget
FGGBAVL_SUM_BUD_ADJT	TotalBudgetAdjustments
FGGBAVL_SUM_YTD_ACTV	TotalYTDActivity
FGGBAVL_SUM_ENCUMB	TotalEncumbrances
FGGBAVL_SUM_BUD_RSRV	TotalBudgetReservations
FGGBAVL_PERIOD	FiscalPeriod

(a) For each of the fields you just added, click in the **As** row on the **Visible Fields (SELECT)** tab, and type in the value listed in the 'Alias' column, above.

e.g., For the field 'FGGBAVL_SUM_ADOPT_BUD,' type 'TotalAdoptedBudget' into the As row on the Visible Fields (SELECT) tab.

(3) Add the following fields to the **Conditional Fields (WHERE)** tab, and add the following variables to the **Condition** row:

Field	Condition
FGGBAVL_COAS_CODE	:main_DD_COA.ChartAcctCode
FGGBAVL_FSyr_CODE	:main_DD_FiscalYr.FiscalYrCode
FGGBAVL_FUND_CODE	:main_LB_Fund.FundCode
FGGBAVL_ORGN_CODE	:main_LB_Orgn.OrganizationCode
FGGBAVL_ACCT_CODE	:main_LB_Acct.AccountCode

(4) Finally, you're going to add one last argument to the **Conditional Fields (WHERE)** tab:

(a) In the empty column, all the way to the right, click in the **And/Or** row at the top of the column and choose 'and'.

(b) The **Table** row will now show '<calculated>'.

(c) Click in the **Field** row, and click on the ellipsis button to open the **SQL Editor** window:

(i) Click the **User-Defined Variable** button (the ABC button on the toolbar) to bring up a list of DataBlock variables:

- 1) Select 'main_BT_RunDashboard'.
- 2) Click the **OK** button.

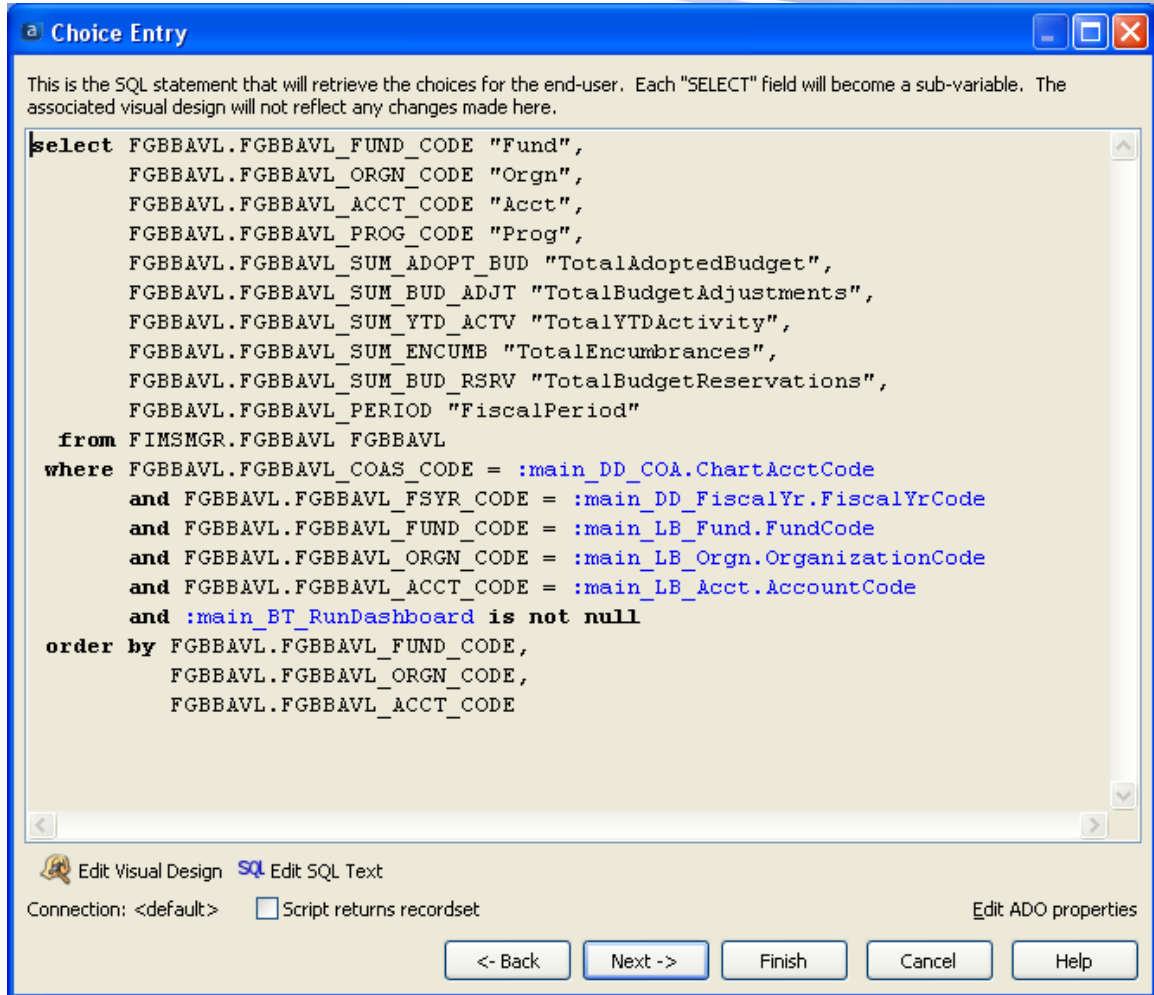
(ii) Click the **OK** button again to get back to the main Visual Editor screen.

(iii) Click in the **Condition** row, and type: 'is not null'

(5) Add the following fields to the **Ordering (ORDER BY)** tab in the order listed: 'FGGBAVL_FUND_CODE', 'FGGBAVL_ORGN_CODE', 'FGGBAVL_ACCT_CODE'

(6) Click the **OK** button to exit the Visual Designer.

- ii. Review the SQL query you just built. It should look like this:



- iii. Click the **NEXT** button.
- iv. Click the **OK** button on the **Test Values** window.
 - REMEMBER: If there is an error in your query, an error message will display, in which case:
 - (a) Read the error message, looking for indications of what might be wrong.
 - (b) Click the **OK** button.
 - (c) Go back to the Visual Designer to find and fix your error. Remember that clicking **Cancel** will cancel all your work.
- v. On the next screen, provided there were no errors, hit the **Finish** button.

Modify the column headings in the multi-column list box

121. Click to select the multi-column list box.

122. On the **Properties** tab:

- a. Next to **Columns**, click the ellipsis button that appears to open the **Edit Columns** window:
 - i. From this screen you can change the text and format of the list box's column headers. The current headings are too long to fit in the columns. To change a column header, select the current header from the list, and type the new name in the **Display Caption** text box. Then type the new column width in the **Width** text box. Make the following changes to your column headers, so everything displays better in the list box:

Old Column Header	New Column Header	New Width
Fund	Fund	80

Orgn	Orgn	80
Acct	Acct	80
Prog	Prog	250
TotalAdoptedBudget	TotalAdoptBudg	120
TotalBudgetAdjustments	TotalBudgAdjust	120
TotalYTDActivity	TotalYTDActiv	120
TotalEncumbrances	TotalEncumb	120
TotalBudgetReservations	TotalBudgReserv	120
FiscalPeriod	FiscalPeriod	80

- ii. Click the **OK** button.

123. Commit and test your changes. (Follow steps 24a-d.)

NOTE: For this DataBlock, try selecting several combinations of values in each of the variables before clicking the Run Dashboard button to display the results in the multi-column list box. Just because your search returns no results doesn't mean the query is broken—try selecting a broad array of values.

Create Report Query for DataBlock by copying and modifying the multi-column list box query

124. Copy the multi-column list box query.

- a. Double-click on the multi-column list box.

- i. Open the Visual Designer.
- ii. Click the **Copy** button in the toolbar.

NOTE: Do not try to copy the Visual Design by using right-click functionality or CTRL-C. You must use the tool in the toolbar.

- iii. Click the **Close** button to exit the Visual Designer.

- b. In the **Choice Entry** window, click the **Cancel** button to return to the main editing window.

125. Create a new Report Query based on the copied SQL.

- a. In the upper left corner, click the **Report Query – Visual Design** tab:

- i. Click the **Paste** button in the toolbar.
- ii. Add tables: 'FTVFUND', 'FTVORGN', 'FTVACCT'

- iii. Create table joins. Each of the three tables you just added needs two joins to the first table—'FGGBAVL'—so for each of the following steps, start in the 'FGGBAVL' table:

- (1) Click on 'FGGBAVL_COAS_CODE' (row 1), drag it to the 'FTVFUND' table window, and drop it on 'FTVFUND_COAS_CODE' (row 1).
- (2) Click on 'FGGBAVL_COAS_CODE' (row 1) *again*, drag it to the 'FTVORGN' table window, and drop it on 'FTVORGN_COAS_CODE' (row 1).
- (3) Click on 'FGGBAVL_COAS_CODE' (row 1) one last time, drag it to the 'FTVACCT' table window, and drop it on 'FTVACCT_COAS_CODE' (row 1).
- (4) Next, click on 'FGGBAVL_FUND_CODE' (row 3), drag it to the 'FTVFUND' table window, and drop it on 'FTVFUND_FUND_CODE' (row 2).
- (5) Click on 'FGGBAVL_ORGN_CODE' (row 4), drag it to the 'FTVORGN' table window, and drop it on 'FTVORGN_ORGN_CODE' (row 2).
- (6) Finally, click on 'FGGBAVL_ACCT_CODE' (row 5), drag it to the 'FTVACCT' table window, and drop it on 'FTVACCT_ACCT_CODE' (row 2).
- (7) You should now have a total of six joins between 'FGGBAVL' and the other 3 tables.

*NOTE: If you accidentally create a join between the wrong two fields, right-click on the join and select **Delete Join** from the menu.*

- iv. Add the following fields to the **Visible Fields (SELECT)** tab, and type the following aliases into the **As** row (if you don't edit the aliases for the last three fields, they'll display as 'Title', 'Title1', and 'Title2', which makes them hard to distinguish from one another):

Table	Field	Alias
FGGBAVL	COAS_CODE	COACode
FGGBAVL	FSYR_CODE	FiscalYrCode
FTVFUND	FTVFUND_TITLE	FundTitle
FTVORGN	FTVORGN_TITLE	OrgTitle
FTVACCT	FTVACCT_TITLE	AcctTitle

- v. Click on the **Conditional Fields (WHERE)** tab:
- (1) Go to the final column, where the ':main_BT_RunDashboard' condition is.
 - (2) Click on the **X** button (next to the arrows at the top of the column) to remove the condition.
- NOTE: The X will turn red when you mouse over it.*
- vi. Click on the **View SQL** button (magnifying glass button on the toolbar), to review your complete Report Query. It should look like this:

```

select FGGBAVL.FGGBAVL_FUND_CODE "Fund",
       FGGBAVL.FGGBAVL_ORGN_CODE "Orgn",
       FGGBAVL.FGGBAVL_ACCT_CODE "Acct",
       FGGBAVL.FGGBAVL_PROG_CODE "Prog",
       FGGBAVL.FGGBAVL_SUM_ADOPT_BUD "TotalAdoptedBudget",
       FGGBAVL.FGGBAVL_SUM_BUD_ADJT "TotalBudgetAdjustments",
       FGGBAVL.FGGBAVL_SUM_YTD_ACTV "TotalYTDActivity",
       FGGBAVL.FGGBAVL_SUM_ENCUMB "TotalEncumbrances",
       FGGBAVL.FGGBAVL_SUM_BUD_RSRV "TotalBudgetReservations",
       FGGBAVL.FGGBAVL_PERIOD "FiscalPeriod",
       FGGBAVL.FGGBAVL_COAS_CODE "COACode",
       FGGBAVL.FGGBAVL_FSYR_CODE "FiscalYrCode",
       FTVFUND.FTVFUND_TITLE "FundTitle",
       FTVORGN.FTVORGN_TITLE "OrgnTitle",
       FTVACCT.FTVACCT_TITLE "AcctTitle"
from FIMSMGR.FGGBAVL FGGBAVL,
     FIMSMGR.FTVFUND FTVFUND,
     FIMSMGR.FTVORGN FTVORGN,
     FIMSMGR.FTVACCT FTVACCT
where ( FGGBAVL.FGGBAVL_COAS_CODE = FTVFUND.FTVFUND_COAS_CODE
       and FGGBAVL.FGGBAVL_COAS_CODE = FTVORGN.FTVORGN_COAS_CODE
       and FGGBAVL.FGGBAVL_COAS_CODE = FTVACCT.FTVACCT_COAS_CODE
       and FGGBAVL.FGGBAVL_FUND_CODE = FTVFUND.FTVFUND_FUND_CODE
       and FGGBAVL.FGGBAVL_ORGN_CODE = FTVORGN.FTVORGN_ORGN_CODE
       and FGGBAVL.FGGBAVL_ACCT_CODE = FTVACCT.FTVACCT_ACCT_CODE )
and ( FGGBAVL.FGGBAVL_COAS_CODE = :main_DD_COA.ChartAcctCode
     and FGGBAVL.FGGBAVL_FSYR_CODE = :main_DD_FiscalYr.FiscalYrCode
     and FGGBAVL.FGGBAVL_FUND_CODE = :main_LB_Fund.FundCode
     and FGGBAVL.FGGBAVL_ORGN_CODE = :main_LB_Orgn.OrganizationCode
     and FGGBAVL.FGGBAVL_ACCT_CODE = :main_LB_Acct.AccountCode )
order by FGGBAVL.FGGBAVL_FUND_CODE,
         FGGBAVL.FGGBAVL_ORGN_CODE,
         FGGBAVL.FGGBAVL_ACCT_CODE

```

126. Commit your changes and test your complete Report Query. *(Follow steps 58a-g.)*

- a. Remember the 'Run Dashboard' button executes the SQL in the multi-column list box. To test the SQL in the Report Query, click the 'Get' button on the testing pane. There is no reason to click the 'Run Dashboard' button to test the SQL in the Report Query.
- b. When you're done testing, click the **Close** button on the testing window.

127. Your second DataBlock is complete. Click the **Close** button again to exit the DataBlock Designer.

Third Exercise: Basic DataBlock 3 – Student Course List

Basic Datablock 3 - Student Courselist

argos
powered by evisions

Retrieve a student's course list

Select Term: 200940 - Summer 2 2009

Enter Student Last Name OR Student ID Number

Last Name:

ID Number:

Select the students from the box below before running reports.

ID Number	Last Name	First Name
0 items		

Include dropped classes in report listing

Products Training Support Feedback www.evisions.com

Exercise Description

This DataBlock allows the user to search the database for a particular student and see a list of students that meet the criteria in a multi-column list box. The user can choose a particular student from the multi-column list box and run a report of that student's course list. In addition to getting more practice with the skills you've been learning, you'll also learn to create conditional groups and several new types of control objects.

Below are descriptions of each of the DataBlock elements you'll be creating along with notes on which database tables they draw from:

Term – drop down box that returns a list of terms. *Table: STVTERM*

Last Name – edit box where a user has the option to search by student last name.

ID Number – edit box where a user has the option to search by student ID number.

You'll also be adding several other elements to the DataBlock, which will add functionality for the end user:

Get List of Students – button that allows the end user to retrieve a list of students that match the first 3 variables. Results are displayed in the Student multi-column list box.

Student List – multi-column list box that will display a list of students, from which the end user can choose the student, or group of students, they need a course listing report for. *Table: SPRIDEN*

Dropped Classes – check box that allows end users to include or exclude dropped classes in the course list.

You'll also be building a Report Query to enable end users to run a course list report for a given student or group of students:

'Student Course List' Report Query – query which displays the course list for selected student or group of students. *Tables: SFRSTCR, STVRSTS, SSBSECT, SPRIDEN*

Instructions

Create a new DataBlock and open it for editing

128. Find your user folder in the **Explorer** tab folder list on the left-hand side.
129. Right-click on your folder, and choose **New > DataBlock** from the menu.
130. Type in the name of your new DataBlock – 'Student Course List' – and hit the **Enter** key.
131. Click on your new DataBlock to highlight it.
132. On the right side of the screen, under the **DataBlock Designer Actions** heading, select the ADO being used for your training from the Associated Connection/Pool drop down.
133. Click the **Edit** button to open the DataBlock Designer.

Configure the form layout and add template objects

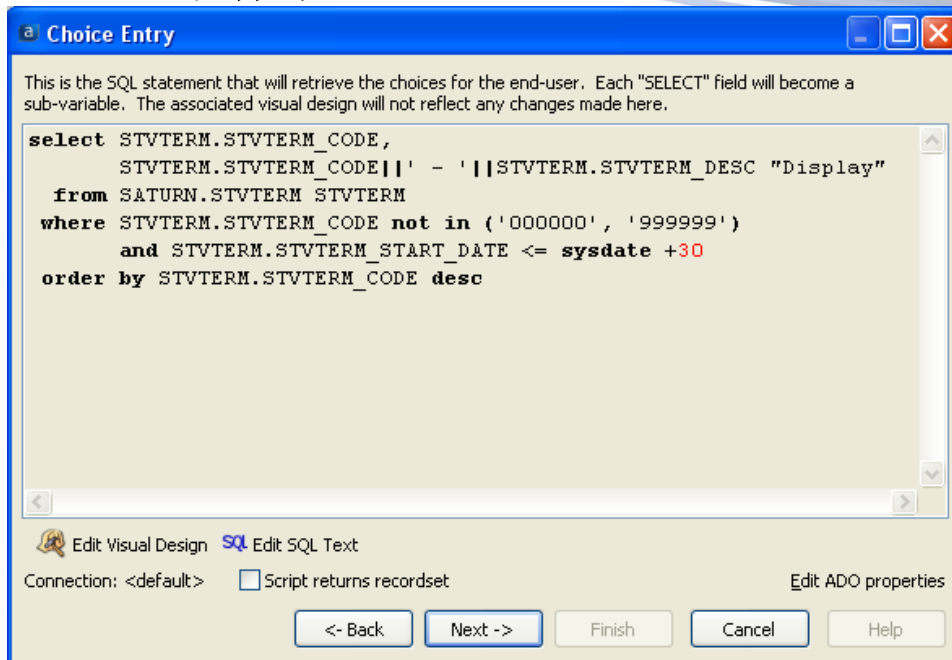
134. Click on the **Properties** tab:
 - a. Next to **Color**, select your background color from the drop down.
135. Click on the **Add Objects from Library** button, to open the **Choose an Object** window:
 - a. Your user folder should be automatically selected. (If it's not, click on it to select it.)
 - b. Click on 'Training DataBlock Template' to select it.
 - c. Click the **OK** button.
136. Double-click on the 'Description/Instructions' label object and change the text to: 'Retrieve a student's course list.'
137. Click the **OK** button.

Add the 'Term' variable

138. Add a label for your variable.
 - a. Add a new label to the panel. *(Follow steps 29-30.)*
 - b. Double-click and change the text to 'Select Term:'.
 - c. On the **Properties** tab:
 - i. Next to **Font.Bold**, choose 'Yes'.
 - d. Click and drag the label to the upper-left corner of the panel.
139. Add a drop down control object.
 - a. Click the **Add Drop Down** tool.
 - b. Click on the panel to place the drop down.
 - c. On the **Properties** tab:
 - i. Next to **Variable Name**, type 'main_DD_Term'.
 - ii. Next to **Auto Select**, choose 'Yes'.
 - d. Click and drag the drop down next to your 'Select Term:' label.
140. Write a SQL query that will supply a list of term codes and descriptions for drop down box.
 - a. Double click on the drop down.
 - i. Open the Visual Designer.
Remember: For more details on working in the Visual Designer, see instructions in steps 90 & 94.

- (1) Add Table: STVTERM
- (2) Add 'STVTERM_CODE' to the **Visible Fields (SELECT)** tab.
- (3) Now to add the calculation to show both the term code and description. On the **Visible Fields (SELECT)** tab:
 - (a) In the furthest column to the right, click in the **Table** row and choose '<calculated>'.
 - (i) Click in the **Field** row and click the ellipsis button to open the **SQL Editor**:
 - (i) Click the **Insert Field** button (the green neutron on the toolbar) to open the **Pick a Field** window:
 - 1) Click the **+** button to expand out 'STVTERM'.
 - 2) Click 'STVTERM_CODE'.
 - 3) Click the **OK** button.
 - (ii) Back in the **SQL Editor**, after 'STVTERM.STVTERM_CODE', type the following:
`|| '-' ||`
REMEMBER: This code will display a hyphen between the two values we're calculating.
 - (iii) Click the **Insert Field** button again, to add the second field to the calculation:
 - 1) Click the **+** button to expand out 'STVTERM'.
 - 2) Click 'STVTERM_DESC'.
 - 3) Click the **OK** button.
 - (iv) Confirm that your string is correct—it should read:
`STVTERM.STVTERM_CODE || '-' || STVTERM.STVTERM_DESC`
 - (v) Click the **OK** button to exit the SQL editor.
 - (c) Click in the **As** row and type 'Display' (instead of 'calc1').
 - (4) Click on the **Conditional Fields (WHERE)** tab to add two conditions:
 - (a) Add 'STVTERM_CODE' to the tab.
 - (b) In the **Condition** row beneath 'STVTERM_CODE', type the following text:
`not in ('000000', '999999')`
 - (c) Add 'STVTERM_START_DATE' to the tab.
 - (d) In the **Condition** row beneath 'STVTERM_START_DATE', type the following text:
`<= sysdate +30`
 - (5) Click on the **Ordering (ORDER BY)** tab:
 - (a) Add 'STVTERM_CODE' to the tab.
 - (b) Click in the **Sort** row, and choose 'Descending'.
 - (6) Click the **OK** button to exit the Visual Designer.

- ii. Review the SQL query you just built. It should look like this:



- iii. Click the **Next** button.
 - iv. You should see your first 5 results. Click the **NEXT** button again.
 - v. Choose 'Display' from the drop down to use your calculated field, and click the **Finish** button.
- b. Commit and test your changes. (Follow steps 24a-d.)

Add a sub-heading for the next two variables

- 141. Add a new label to the panel. (Follow steps 29-30.)
- 142. Double-click and change the text to 'Enter Student Last Name OR Student ID Number'.
- 143. On the **Properties** tab:
 - a. Next to **Font.Bold**, choose 'Yes'.
 - b. Click and drag the sub-heading under the 'Select Term' variable.

Add the 'Last Name' variable

- 144. Add a label for your variable.
 - a. Add a new label to the panel. (Follow steps 29-30.)
 - b. Double-click and change the text to 'Last Name:'.
 - c. On the **Properties** tab:
 - i. Next to **Font.Bold**, choose 'Yes'.
 - d. Click and drag the label to the left side of the panel, under the sub-heading.
- 145. Add an edit box control object.
 - a. Click on the **Add Edit Box** tool (the button on the tool bar with 'ab' and a cursor on it).



- b. Click on the panel to add the edit box.
- c. On the **Properties** tab:
 - i. Next to **Required**, choose 'No'.
 - ii. Next to **Variable Name**, type 'main_EB_LastName'.

- d. Click and drag the Edit Box next to your 'Last Name:' label.

Add the 'ID Number' variable

146. Add a label for your variable.
 - a. Add a new label to the panel. *(Follow steps 29-30.)*
 - b. Double-click and change the text to 'ID Number:'.
 - c. On the **Properties** tab:
 - i. Next to **Font.Bold**, choose 'Yes'.
 - d. Click and drag the label to the left side of the panel, under the 'Last Name' edit box.
147. Add an edit box control object.
 - a. Click on the **Add Edit Box** tool (the button on the tool bar with 'ab' and a cursor on it).
 - b. Click on the panel to add the edit box.
 - c. On the **Properties** tab:
 - i. Next to **Required**, choose 'No'.
 - ii. Next to **Variable Name**, type 'main_EB_IDNumber'.
 - d. Click and drag the Edit Box next to your 'ID Number:' label.

Add the 'Get List of Students' button

148. Click the **Add Button** tool (the OK button in the toolbar).
149. Click on the panel to add the button.
150. On the **Properties** tab:
 - a. Next to **Variable Name**, type 'main_BT_RunDashboard'.
 - b. Next to **Font.Bold**, choose 'Yes'.
 - c. Next to **Tab Stop**, choose 'Yes'.
 - d. Next to **Text**, type 'Get List of Students'.
151. Click and drag one of the button's corners to resize to fit the text.
152. Click and drag button to position it in the middle of the form, under the edit boxes.

Add a sub-heading for the multi-column list box

153. Add a new label to the panel. *(Follow steps 29-30.)*
154. Double-click and change the text to 'Select students from the box below before running reports.'
155. On the **Properties** tab:
 - a. Next to **Font.Bold**, choose 'Yes'.
156. Click and drag the sub-heading to the left, under the 'Get List of Students' button.

Add the 'Student List' multi-column list box to display student information

157. Add multi-column list box control object.
 - a. Click on the **Add Multi-Column List** Box tool (the button on the toolbar that shows a list with two columns).
 - b. Click in the panel to place the multi-column list box.
 - c. Click and drag one of multi-column list box's corners to increase its size until it takes up about two-thirds of the remaining space on the bottom of the panel, leaving room on the right.
 - d. On the **Properties** tab:
 - i. Next to **Auto Select**, choose 'Yes'.
 - ii. Next to **Multi Select**, choose 'Yes'.
 - iii. Next to **Show Item Count**, choose 'Yes'.
 - iv. Next to **Variable Name**, type 'main_MC_StudentList'.

158. Write a SQL Query to return the student data.

a. Double-click on the multi-column list box.

i. Open the Visual Designer.

Remember: For more details on working in the Visual Designer, see instructions in steps 90 & 94.

(1) Add Table: SPRIDEN

(2) Add the following fields to the **Visible Fields (SELECT)** tab: 'SPRIDEN_ID', 'SPRIDEN_LAST_NAME', 'SPRIDEN_FIRST_NAME', 'SPRIDEN_PIDM'

(3) Add the following fields to the **Conditional Fields (WHERE)** tab, and add the following arguments to the **Condition** row:

Field	Condition
SPRIDEN_SEARCH_LAST_NAME	like upper (:main_EB_LastName '%')
SPRIDEN_ID	like upper (:main_EB_IDNumber '%')
SPRIDEN_ENTITY_IND	= 'P'

(a) Add a new condition:

(i) In the furthest column to the right, click in the **And/Or** row at the top of the column and choose 'and'.

NOTE: Selecting 'and' here means that the value for the Table row will automatically be '<calculated>', so you won't have to enter anything in that row.

(ii) In the **Field** row, click the ellipsis button to open the **SQL Editor** window:

1) Type in the following text: f_registered_this_term(

2) Click on the **Insert Field** button (the green neutron button on the toolbar) to open the **Pick a Field** window:

a) Click the + button to expand out the 'SPRIDEN' list.

b) Select 'SPRIDEN_PIDM'.

c) Click the **OK** button.

3) Type: ,

4) Click the **User-Defined Variable** button (ABC button) to bring up a list of DataBlock variables:

a) Find and select 'main_DD_term.TermCode'.

b) Click the **OK** button.

5) Type:)

6) Your SQL should now say:

f_registered_this_term(SPRIDEN.SPRIDEN_PIDM,;main_DD_term.TermCode)

(iii) Click the **OK** button

(iv) In the **Condition** row, type the following:

= 'Y'

(b) And one last condition:

(i) In the furthest column to the right, click in the **And/Or** row at the top of the column and choose 'and'.

(ii) In the **Field** row, click the ellipsis button to open the **SQL Editor** window:

1) Click the **User-Defined Variable** button (ABC button) to bring up a list of DataBlock variables:

a) Find and select 'main_BT_RunDashboard'.

b) Click the **OK** button.

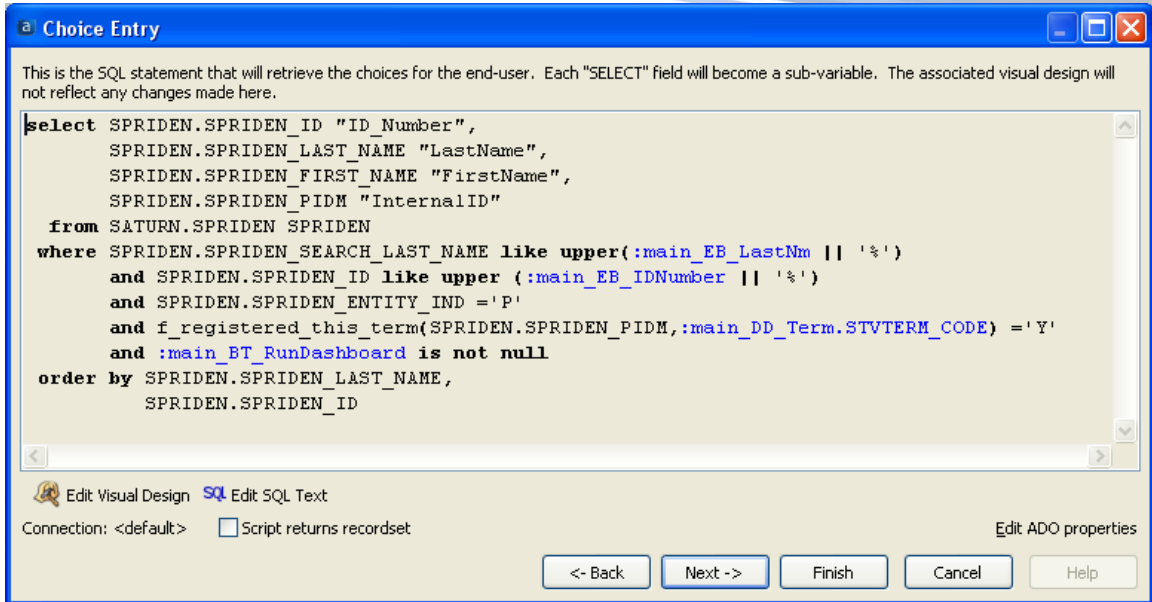
2) In the **Condition** row, type the following:

is not null

(4) Add the following fields to the **Ordering (ORDER BY)** tab: 'SPRIDEN_LAST_NAME' and 'SPRIDEN_ID'

(5) Click the **OK** button to exit the Visual Designer.

- ii. Review the SQL query you just built. It should look like this:



- iii. Click the **NEXT** button. The **Test Values** window will appear, asking you to supply test values for the variables on the form. Leave everything blank and hit the **OK** button. (You'll do a test run in a minute.)
 - iv. Use the Visual Designer to correct errors, if any, and then hit the **Finish** button.
- b. Commit and test your changes. (Follow steps 24a-d.)

Modify the column headings in the multi-column list box

- 159. Click to select the multi-column list box.
- 160. On the **Properties** tab:

- a. Next to **Columns**, click the ellipsis button that appears to open the **Edit Columns** window:
 - i. For this DataBlock, we don't need to have all the column headers visible, and we also have the room to make them a little more readable. To change a column header, select the current header from the list, and type the new name in the **Display Caption** text box. Then type the new column width in the **Width** text box. Make the following changes to your column headers:

Old Column Header	New Column Header	New Width
ID_Number	ID Number	150
LastName	Last Name	200
FirstName	First Name	200
InternalID -- This column isn't useful for the end users, so we're not going to display it to them. Uncheck the Visible checkbox.		

- ii. Click the **OK** button.

- 161. Commit and test your changes. (Follow steps 24a-d.)

Add the 'Dropped Classes' checkbox

- 162. Click the **Add Check Box** tool (the button in the toolbar that shows a box with an 'x' in it)



- 163. Click on the panel to add the button.
- 164. On the **Properties** tab:

- a. Next to **Checked**, choose 'Yes'.
- b. Next to **Checked Value**, type 'Y'.
- c. Next to **Font.Bold**, choose 'Yes'.
- d. Next to **Text**, type 'Include dropped classes in report listing?'.
- e. Next to **UnChecked Value**, type 'N'.
- f. Next to **Variable Name**, type 'main_CB_IncludeDropped'.

165. Click and drag one of the variable's corners to resize to fit the text.

166. Click and drag the check box to the right of the multi-column list box.

167. Commit and test your changes. (Follow steps 24a-d)

Create a Report Query for this DataBlock

168. In the upper left corner, click the **Report Query – Visual Design** tab:

Remember: For more details on working in the Visual Designer, see instructions in steps 90 & 94.

- a. Add tables: 'SFRSTCR', 'STVRSTS', 'SSBSECT', 'SPRIDEN'
- b. Create table joins. The 'SFRSTCR' table needs to be joined to each of the other three tables. For the following steps, start in the 'SFRSTCR' table window:
 - i. Click on 'SFRSTCR_RSTS_CODE' (row 7), drag it to the 'STVRSTS' table window, and drop it on 'STVRSTS_CODE' (row 1).
 - ii. Click on 'SFRSTCR_TERM_CODE' (row 1), drag it to the 'SSBSECT' table window, and drop it on 'SSBSECT_TERM_CODE' (row 1).
 - iii. Click on 'SFRSTCR_CRN' (row 3), drag it to the 'SSBSECT' table window, and drop it on 'SSBSECT_CRN' (row 2).
 - iv. Click on 'SFRSTCR_PIDM' (row 2), drag it to the 'SPRIDEN' table window, and drop it on 'SPRIDEN_PIDM' (row 1).
 - v. You should now have a total of four joins between 'SFRSTCR' and the other three tables.
- c. Add the following fields to the **Visible Fields (SELECT)** tab:

SFRSTCR	STVRSTS	SSBSECT	SPRIDEN
SFRSTCR_CRN	STVRSTS_DESC	SSBSECT_SUBJ_CODE	SPRIDEN_ID
SFRSTCR_GRDE_CODE		SSBSECT_CRSE_NUMB	SPRIDEN_LAST_NAME
SFRSTCR_CREDIT_HR			SPRIDEN_FIRST_NAME
SFRSTCR_RSTS_CODE			

- d. Add the following fields to the **Conditional Fields (WHERE)** tab, and add the following arguments to the **Condition** row:

Table	Field	Condition
SFRSTCR	SFRSTCR_PIDM	=:main_MC_StudentList.InternalID
SFRSTCR	SFRSTCR_TERM_CODE	=:main_DD_Term.STVTERM_CODE

- i. In the lower left-hand corner of the **Conditional Fields (WHERE)** tab, click on the **+** button (it will turn green when you mouse over it) to create a conditional group:
 - (1) In the list on the left, you'll see a new '<group>' beneath your root query with a new set of empty columns to add new conditions.
NOTE: To go back to your root conditions, simply click on '<root>'.
 - (2) Double-click 'SFRSTCR_RSTS_CODE', in the 'SFRSTCR' table window to add it to your conditional group.
 - (a) Click in the **Condition** row, type the following text:

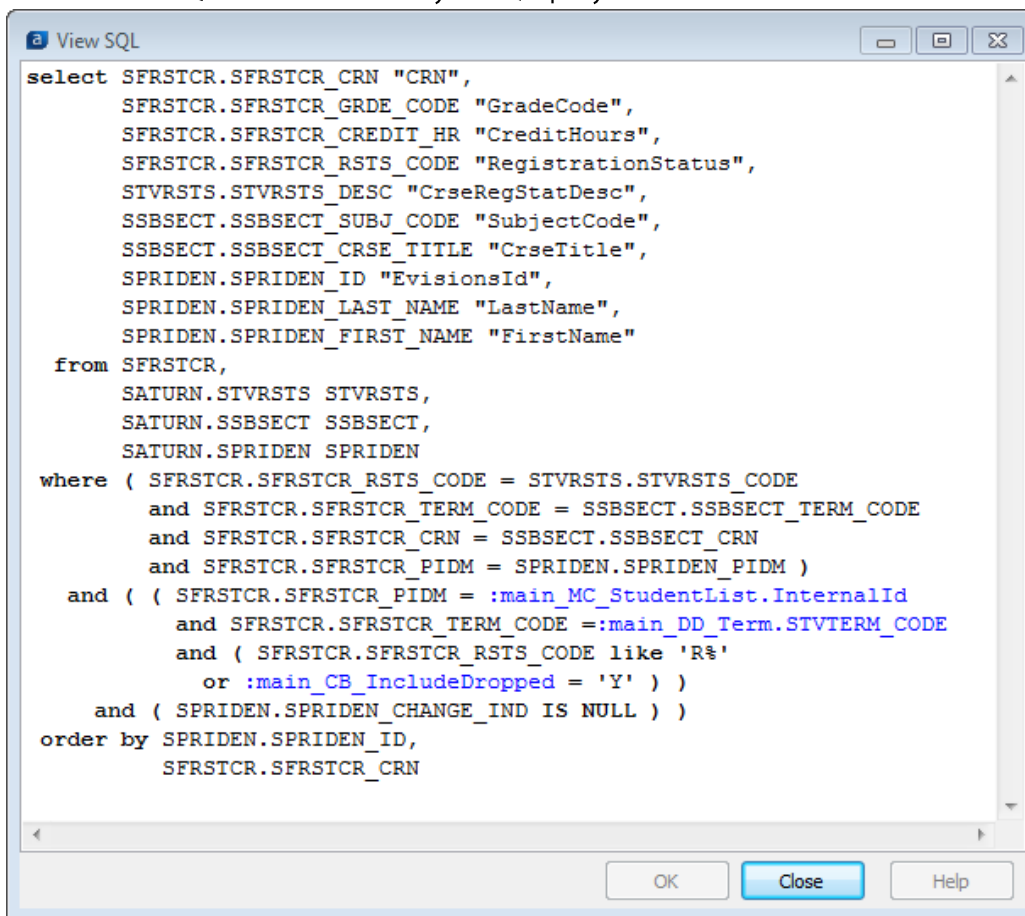
like 'R%'

- (3) Still in your conditional group, in the furthest column to the right, add one more condition:
 - (a) Click in the **And/Or** row at the top of the column and choose 'or'.
NOTE: Be sure to choose 'or' here, not 'and' as you've been doing up until now.
 - (b) In the **Field** row, click the ellipsis button to open the **SQL Editor** window:
 - (i) Click the **User-Defined Variable** button (ABC button) to bring up a list of DataBlock variables:
 - 1) Find and select 'main_CB_IncludeDropped'.
 - 2) Click the **OK** button.
 - (c) In the **Condition** row, type the following:
='Y'

- e. Add the following fields to the **Ordering (ORDER BY)** tab:

Table	Field
SPRIDEN	SPRIDEN_ID
SFRSTCR	SFSTCR_CRN

- f. Click the **View SQL** button and review your SQL query. It should look like this:



```
select SFRSTCR.SFRSTCR_CRN "CRN",
       SFRSTCR.SFRSTCR_GRDE_CODE "GradeCode",
       SFRSTCR.SFRSTCR_CREDIT_HR "CreditHours",
       SFRSTCR.SFRSTCR_RSTS_CODE "RegistrationStatus",
       STVRSTS.STVRSTS_DESC "CrseRegStatDesc",
       SSBSECT.SSBSECT_SUBJ_CODE "SubjectCode",
       SSBSECT.SSBSECT_CRSE_TITLE "CrseTitle",
       SPRIDEN.SPRIDEN_ID "EvisionsId",
       SPRIDEN.SPRIDEN_LAST_NAME "LastName",
       SPRIDEN.SPRIDEN_FIRST_NAME "FirstName"
from SFRSTCR,
     SATURN.STVRSTS STVRSTS,
     SATURN.SSBSECT SSBSECT,
     SATURN.SPRIDEN SPRIDEN
where ( SFRSTCR.SFRSTCR_RSTS_CODE = STVRSTS.STVRSTS_CODE
       and SFRSTCR.SFRSTCR_TERM_CODE = SSBSECT.SSBSECT_TERM_CODE
       and SFRSTCR.SFRSTCR_CRN = SSBSECT.SSBSECT_CRN
       and SFRSTCR.SFRSTCR_PIDM = SPRIDEN.SPRIDEN_PIDM )
       and ( ( SFRSTCR.SFRSTCR_PIDM = :main_MC_StudentList.InternalId
              and SFRSTCR.SFRSTCR_TERM_CODE =:main_DD_Term.STVTERM_CODE
              and ( SFRSTCR.SFRSTCR_RSTS_CODE like 'R%'
                    or :main_CB_IncludeDropped = 'Y' ) )
           and ( SPRIDEN.SPRIDEN_CHANGE_IND IS NULL ) )
order by SPRIDEN.SPRIDEN_ID,
         SFRSTCR.SFRSTCR_CRN
```

169. Test your complete Report Query. (Follow steps 58a-g.)

- a. You can test out your form by selecting a term and filling in the first letter or two of a common last name.
- b. When you're done testing, click the **Close** button on the testing window.

170. Your third DataBlock is complete. Click the **Close** button again to exit the DataBlock Designer.

Appendix I: DataBlock Naming Conventions

Using a naming convention to keep track of all of the different elements of a DataBlock is incredibly useful (when looking for a particular element in a list, troubleshooting a complex SQL query, etc.).

If your institution already has naming conventions you use, please feel free to follow them when naming variables in these exercises. (And let your trainer know, too, so they can get everyone in the class on the same page.)

If your institution does not already have naming conventions, there's no better time to get into the habit than now. Below, you'll find some suggested conventions, as well as a list of useful abbreviations:

Naming Convention for Form Variables

[FormName]_[AbbreviatedObjectType]_[VariableDescription]

For example: *main_DD_FiscalYr* would refer to a variable on the *main* form, with a *drop down (DD)* control object that contains a list of *Fiscal Years (Yr)*.

The above convention ensures that just by looking at the variable's name, you can infer the location of the variable, the type of control object it is, and a brief description of the data the variable contains. For abbreviating the type of control object, use the codes below:

- QB - OLAP Cube
- MC - Multi-Column List Box
- LB - Single Column List Box
- DD - Drop Down Control
- CT - Chart
- BT - Button
- CB - Check Box
- RB - Radio Button
- DT - Date Control
- EB - Edit Box
- MB - Memo Box
- PL - Panel
- LL - Labels
- SP - Shape
- IM - Image
- SB - Scroll Box


Naming Convention for SQL Variables:

[FormName]_[ObjectType]_[VariableDescription]

For example: *main_SQL_GetName* would refer to a SQL variable on the *main* form, in the form of a *SQL Select Statement* that *returns names*.

Here are a couple of useful abbreviations for naming SQL variables:

- SQL - SQL Select Statement
- PLSQL - Anonymous Block
- MAN - Manual Entry Variable



Trademark, Publishing Statement and Copyright Notice

© 2013 Evisions, Inc. All rights reserved.

This software and related documentation are provided under a license agreement containing restrictions on use and disclosure and are protected by intellectual property laws. No part of this document may be reproduced, stored in or introduced into a retrieval system, or transmitted in any form or by any means (electronic, mechanical, photocopying, recording, or otherwise), or for any purpose, without the express written permission of Evisions, Inc.

The information contained herein is subject to change without notice and is not warranted to be error-free. Product features referenced herein for a period of time may not match product contents. Evisions, Inc. does not warrant that the functions contained in the software will meet your requirements or that the operation of the software will be uninterrupted or error free. Evisions, Inc. reserves the right to make changes and/or improvements in the software without notice at any time.

This software and documentation may provide access to or information on content, products, and services from third parties. Evisions, Inc. and its affiliates are not responsible for and expressly disclaim all warranties of any kind with respect to third-party content, products, and services. Evisions, Inc. and its affiliates will not be responsible for any loss, costs, or damages incurred due to your access to or use of third-party content, products, or services. Evisions, Inc. does not endorse the content or developer of any products or web sites mentioned.

All information in this guide is designed for instructional purposes only. Evisions, Inc. makes no guarantees regarding the accuracy or performance of any techniques used in this guide. Software configurations and environments may vary, and some techniques used in this guide may not operate efficiently under all configurations. This guide may contain examples of various technologies or products, which are the sole property and responsibility of their creators.

Trademarks are the property of the respective owners for any products mentioned herein.