

FormFusion Suite

Working with Student Invoice

The following is a brief guide to how the sample student bill **INVOICE** was created. We discuss how to: (1) **search for specific data in Banner output and capture it**, (2) **delete captured information**, (3) **reuse it elsewhere**, (4) **perform a calculation based upon captured data**, (5) **selectively print different forms based upon conditional logic**. Refer to Appendix A for Before and After.

Client Requirements:

1. The client wanted a simple student invoice statement that they could implement rapidly. However, they wanted the total fields to be printed in certain areas of the statement and to include a 'tear-off' portion that the students could return/remit with payment. In addition to moving these fields, they did not want to print the dashed lines and messages that comes from Banner baseline.
2. The due date would be calculated as 30 days from the date the bill was printed. They also wanted the term description to print rather than the term code.
3. If a printout was over one page in length, only the final page of the invoice would contain any information reflecting the amount due. We will use overlays to accomplish this.

Starting the Process:

With beginning the design process, it is helpful to import a template that closely matches what you are working towards. By simply importing/opening your sample template (in this case the `tsrcbil_invoice.ffe`) you will be able to see what we did to meet the clients needs and be able to make your own modifications. Please refer to appendix B to see the details on how we got to this point.

1. Using Offset fields in MapForm

When the TSRCBIL is executed in INVOICE mode, the total line (PAST DUE, FUTURE BALANCE and TOTAL DUE fields) prints immediately after the last detail line is printed. i.e., the total line can appear anywhere on the page. We want to somehow determine where the information is printing on the list file and assign it to a **MapForm** variable. We do this by using the "offset field."

An offset field should be used when a simple "floating field" will not suffice. This type of field will search for key text information in the report and allow you to map a field from the top-left of where the key text was discovered.

In addition to moving the found information to a fixed spot on the printed output, the surrounding lines that contain dashes and message needed to be removed. Using an offset field also did this.



This is the **Draw Offset Field** tool button.

Click this tool button to create an offset field on the form. You will first be prompted to enter a variable name. Enter the name you want to associate with this field and click OK.

The Offset field properties form will then be displayed. On this form you will enter the word or set of characters you are searching for and where on the page this string could be found. Finally, if found, where is the data that you want to map. For the Total Due field, we will search for the characters "TOTAL" between columns 55 and 59 and rows 1 to 57. If the search text is found, the data we are interested in is 11 columns to the left of the first character of the string being searched for and is 12 characters long. (Refer to figure 1).

For more information on using the offset field, please refer to the **Help: Offset Field Properties**

Although we want the data to NOT print where it currently is printing in the list file, we have selected to leave the text in the report. To actually remove the total line from the report, we are going to define another Offset Variable called Z-DASHES. (Refer to Figure 2).

The string we are searching for is a string of 8 dashes between columns 1 and 8 and rows 21 and 60. We had to begin the search below row 20 so that the dashed line below the column titles would not be the one found. In looking for a field, the search starts on the first row indicated and seeks downward until the first occurrence of the text is found.

Notice that we are not concerned with the column offset, as we really have no intention of printing the return value. However, take note that the row offset is from 0 (the row on which the text is found) and is five rows high – the current row plus the next four rows.

The final thing to note is that the "Remove entire line from report" radio button is checked. This will remove the mapped area and all text to the right and left of the mapped area. In addition, it will remove the five rows associated with the offset.

The screenshot shows the 'Offset Field Properties' dialog box. The 'Name' field contains 'TotalDue'. The 'Field Type' is set to 'String' and the 'Description' is 'Total Due'. Under 'Search for ...', the 'Text' is 'TOTAL'. The search range is defined as 'Between column 55 and column 59' and 'Between row 1 and row 57'. The 'If found, map field using ...' section has 'Column Offset' set to 11, 'Column Width' set to 12, 'Row Offset' set to 0, and 'Row Height' set to 1. The radio button 'Do nothing (leave text in the report)' is selected. At the bottom, there are buttons for 'OK', 'Close', 'Apply', and 'Help'.

Figure 1 – Offset Field Properties

The screenshot shows the 'Offset Field Properties' dialog box. The 'Name' field contains 'Z Dashes'. The 'Field Type' is set to 'String' and the 'Description' is empty. Under 'Search for ...', the 'Text' is '-----'. The search range is defined as 'Between column 1 and column 8' and 'Between row 21 and row 60'. The 'If found, map field using ...' section has 'Column Offset' set to 0, 'Column Width' set to 1, 'Row Offset' set to 0, and 'Row Height' set to 5. The radio button 'Remove entire line from report' is selected. At the bottom, there are buttons for 'OK', 'Close', 'Apply', and 'Help'.

Figure 2 Offset Field Properties - Z-DASHES

2. Due Date and Term Description

CaptureFormSQL is a module within FormFusion that performs a variety of important functions. The primary use of CaptureFormSQL is to retrieve additional information from your Oracle environment for use in your document output. However, there are a number of other uses for CaptureForm as well. These include items such as mathematical calculations, inserting or deleting information in your Oracle environment and formatting character strings. With all CaptureFormSQL statements you can use "fields" from the MapForm module to aid in your selection criteria within the SQL statement. Thus, you have the ability to use information on the document to tie back to various pieces of Banner/database information stored elsewhere.

A CaptureForm is created as a child node of a Special Print Parameter (SPP). From the tree view, select the SPP for which you would like to create a CaptureForm. On the "Edit" menu, point to "New" and then click "CaptureForm". Only one CaptureForm can be added to an SPP; however, unlimited queries and variables can be created within each CaptureForm process modifier.

The first piece of data that we want to retrieve is the *Due Date*. It was decided by the client that this would always be the date the process is run + 30. We must first define a variable called *due_date*. To define a new variable, click on the Variable Storage icon in Capture form and enter the required information. Fig. 3 shows the **Variable Storage Dialog**. In this dialog you can manage the variable names owned by this CaptureForm.

Figure 3 – defining the DUE_DATE variable in CaptureFormSQL.

Variable Name: This is where you can type the name of the variable.

Description: This area allows you to type a short description for this variable. This should describe the general purpose of the variable or what it will be used for. FormFusion never uses the description and is for documentation only.

Data Type: This drop-down combo box allows you to specify the expected format of the variable. If the specified format of the variable differs from the actual format retrieved from the database, then the data

will be converted before the variable is populated.

Visual Settings (for use in FormStamp): The two fields in this area "Estimated Width" and "Estimated Height" are used to estimate the maximum area the variable will take up if ever placed

onto a **FormStamp** PCLForm. These values do not limit the amount of data retrieved from the database.

After the variable is defined, it can be used in a **CaptureFormSQL** query. To add a new query, on the tool bar click on "Edit" from the menu and choose "New" then "CaptureForm Query". A default query node will be created. You are now able to edit the properties of the query.

Edit a Query: Double-click the CaptureForm Query node to edit. A SQL Builder window will open which will allow you to modify the properties of the query.

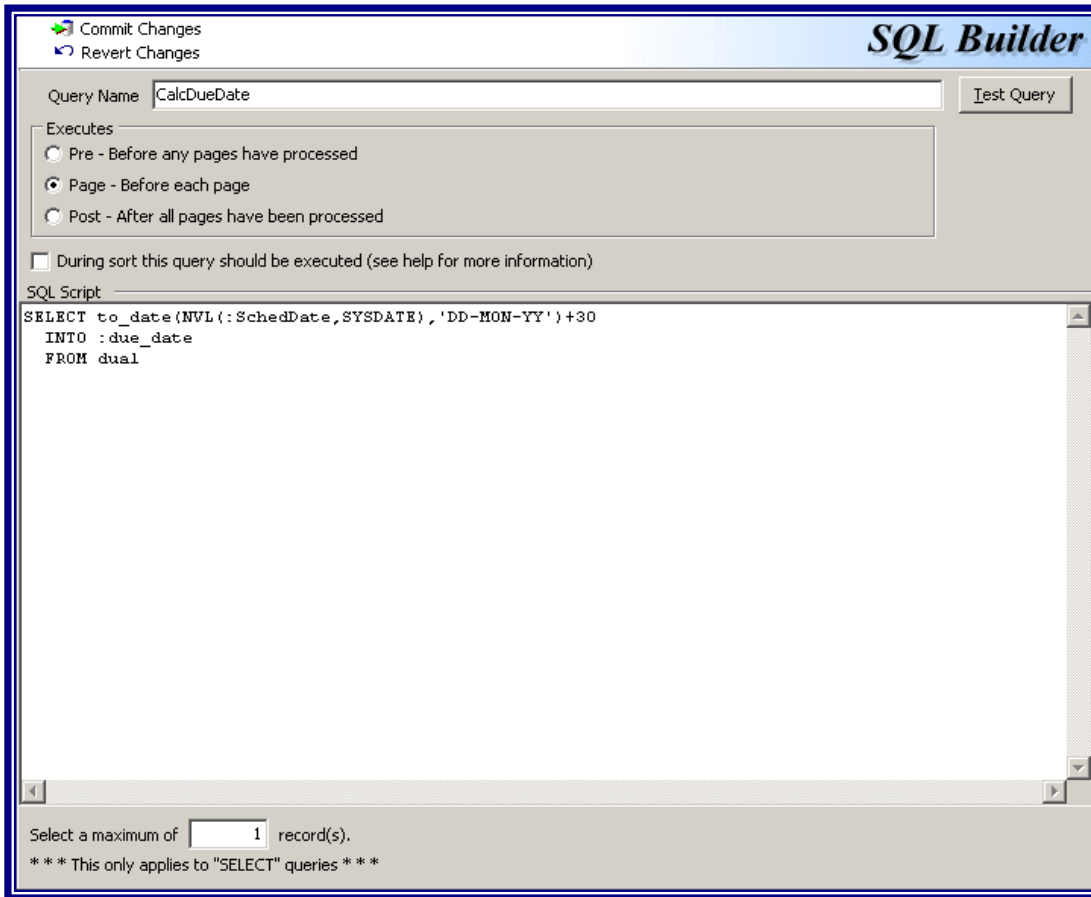


Figure 4 – Sample SQL Builder screen

During sort this query should be executed: To facilitate the sorting process you must specify which queries should be executed during the sort. This decreases program latency by only executing the queries that you instruct FormFusion to execute. Sorting is performed before the processing of the report file has begun. See **Help: Special Print Parameter, SORTING** for more information.

SQL: Below the "Executes" area you will find an edit box. This box is where the SQL query can be created that defines the actions for the CaptureForm Query node. Any standard SQL statement can be entered here. This includes the statements SELECT, INSERT INTO, UPDATE, and DELETE. Variables can be used almost anywhere in the query. The most common use for variables are in the INTO clause of a SELECT statement or in the WHERE clause. In Figure 4 we are using a simple select clause to retrieve or calculate the due date.

Query Name: This is the name of the query as it appears in the tree view. The query name isn't used by FormFusion but is provided to allow you to create a descriptive name for the query.

Executes: This area controls when the query will execute. "Pre" queries execute before any data has been processed on the server side. "Page" queries execute before each page is processed. "Post" queries execute after the entire job is complete.

Select a maximum of # record(s): This edit box allows you to specify the maximum number of records to retrieve and place into the variables in the INTO clause of a SELECT statement. This field is only used for SELECT statements. This is useful in limiting the number of rows that will be selected if you expect more than one value to be returned.

Test Query: The Test Query button allows you to test the query you created without committing the results. You will be prompted for the values of all variables in the query. This is useful for testing the syntax, but is no guarantee that the proper value will be returned.

Commit Changes: After making any changes to the query properties an option will appear at the top of the window to allow you to commit your changes to the database. Any changes that are made are not saved to the database until you click this button. Similarly, if you have made a mistake you can click the "Revert Changes" button, which will revert all changes since the last time you committed. A similar query was written to retrieve the Term Description from the STVTERM table.

3. Use of overlays to selectively print different forms

The TSRCBIL, when run in Invoicing mode, can produce multi-page bills. We want the Total amounts to only print on the last page as having one page indicate Amount Due of \$0.00 and another page indicating that \$2000.00 is owed could be confusing!

To accomplish this, two PCL forms were created in **FormStamp**. The first, called **StudentBill** contains all the mapped fields from **MapForm** and **CaptureFormSQL**. Note that we are also mapping the total fields on this form! However, we are not including the title to these fields. (Note the TotalDue variable in Figure 5).

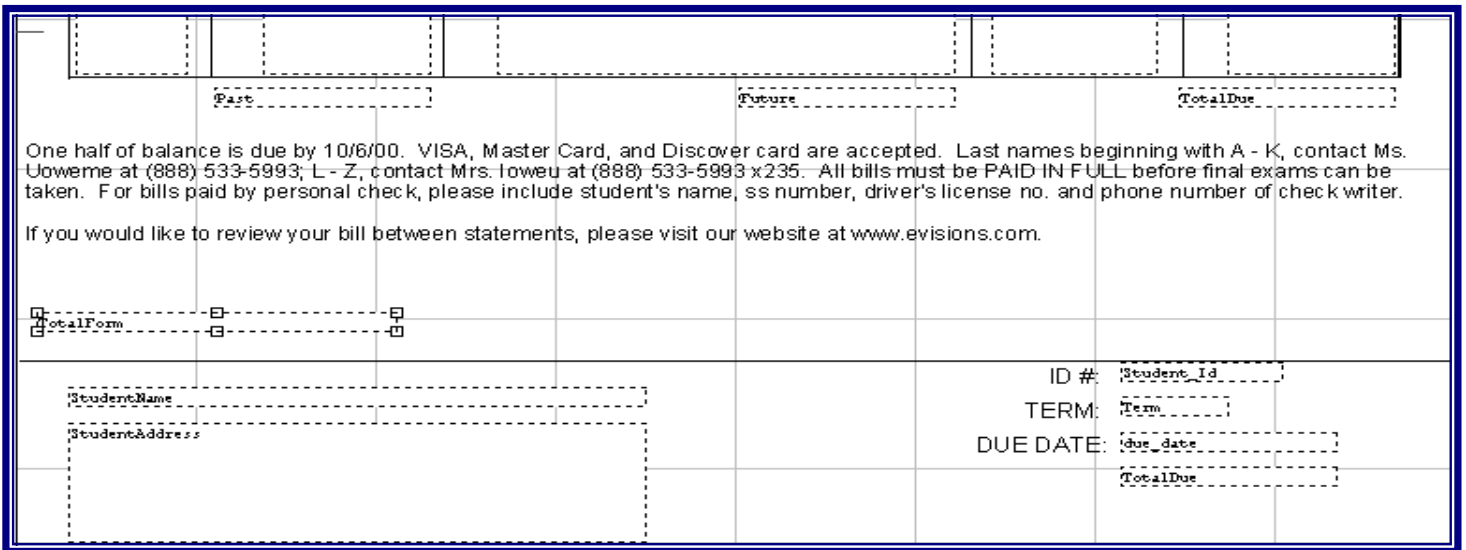


Figure 5 – Bottom portion of StudentBill in FormStamp

We want this form or template to print on every page. In the Form Properties, make sure that the parameter asking for 'Prints on copies' is left blank. Refer to the **HELP:FormStamp – Form Properties** for more information on this.

The next form we create is called **TotalLines**. This form will contain ONLY the information that needs to be added to the StudentBill form IF this is the last page (or only page) of the student's Invoice.

PAST DUE:		FUTURE BALANCE:		TOTAL DUE:	
PLEASE CUT ALONG THE LINE AND RETURN THE PORTION BELOW WITH PAYMENT					
				AMOUNT DUE \$ _____	
				AMOUNT ENCLOSED \$ _____	


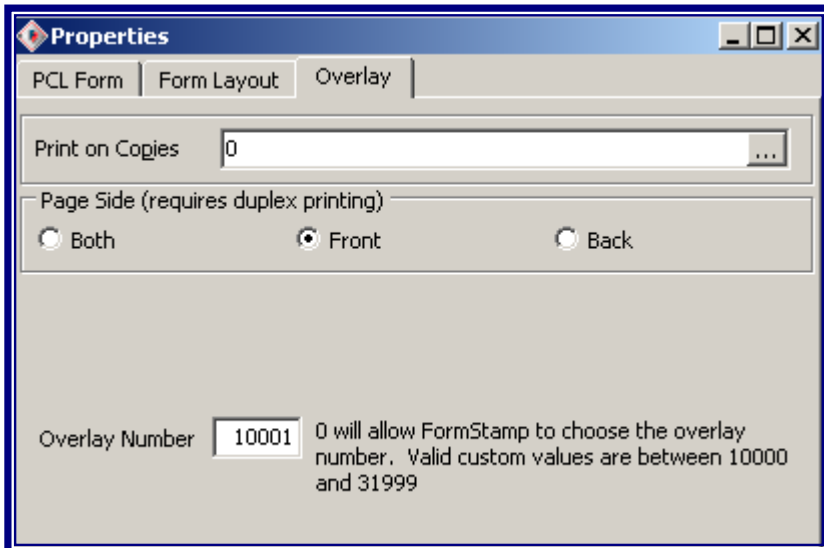


Figure 6 – Bottom of TotalLines in FormStamp

Now the object is to tell FormFusion NOT to print this UNLESS there is something in the TotalDue variable that was mapped in MapForm as one of the offset variables. The first thing that needs to be done is to set the Form properties as shown if Figure 7. The first thing to take note of is that we have entered the value of zero (0) in the Print On Copies field, indicating that this form is NEVER printed. However, we have also assigned an Overlay number of 10001.



Properties

PCL Form | Form Layout | **Overlay**

Print on Copies:

Page Side (requires duplex printing):
 Both Front Back

Overlay Number: 0 will allow FormStamp to choose the overlay number. Valid custom values are between 10000 and 31999

Figure 7 – Overlay tab of Form Properties for TotalLines form

An overlay number is a PCL term. Also known as a macro, the overlay contains all the graphics, text, lines, etc. that will print for the form. FormFusion will automatically number macros sequentially starting with 1 and stopping before 10000. If you would like to take control of the macro number, you may choose a number between 10000 and 31999. By assigning a value of 10001 as the overlay number, we have created an overlay or macro that we can call if certain conditions are met, as in the

TotalDue value not being null.

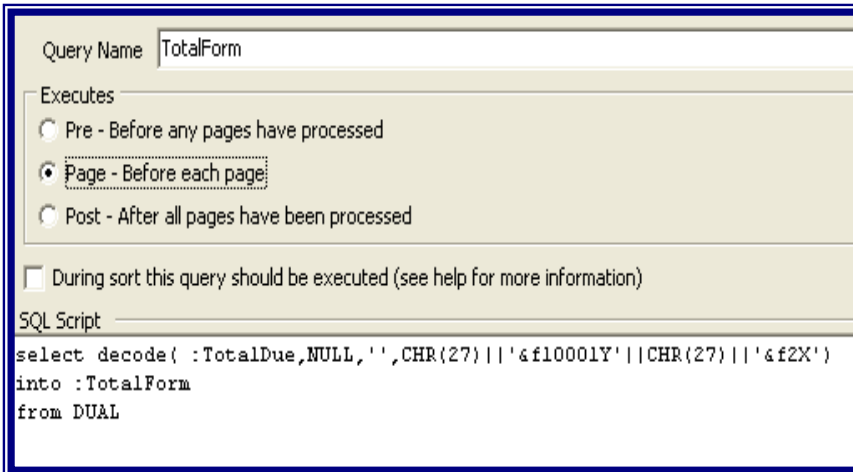
Refer back to Figure 5 you will see a field named **TotalForm**.

In CaptureForm we have created the following query shown in Figure 8. We use the SQL “Decode” command, which is a handy value-substitution mechanism. Here we are assigning the macro number to **TotalForm** IF the **TotalDue** value is NOT NULL, otherwise, the **TotalForm** gets a value of blank.

For example, if this were a two-page bill, the first page, where the **TotalDue** variable would be NULL, then the **TotalForm** variable will be blank. On the second page, the **TotalDue** value will have a value and therefore the **TotalForm** will be assigned the appropriate Macro or overlay number of 10001.

FormFusion, when processing the output, will print the additional template **TotalLines** only if there is a value in the **TotalForm** variable.

NOTE: This is an advance concept. Please refer to the HELP system for more information. If you are using your own macros to determine what forms get printed when, you will need to make sure that the special escape characters in front of and behind the overlay number are COPIED from this query or one like this.



The screenshot shows a query configuration window with the following details:

- Query Name:** TotalForm
- Executes:** Radio buttons for execution timing: Pre - Before any pages have processed, Page - Before each page:, Post - After all pages have been processed.
- During sort this query should be executed (see help for more information)
- SQL Script:**

```
select decode( :TotalDue,NULL,'',CHR(27)||'&f10001Y'||CHR(27)||'&f2X')
into :TotalForm
from DUAL
```

Figure 8 – SQL statement to see what overlay to print