

MODULE 6: REPORTBUILDER AND QUERYDOCUMENTNODESET

In past modules we have seen how DBXL can map data to a SQL database, and how QueryDB can query data from a SQL database.

What if you want to query form data DIRECTLY from the XML forms? QueryDocumentNodeSet is a web service included in DBXL, and Report Builder is its interface. With ReportBuilder you'll be able to create a report that extracts data from the XML files. With QueryDocumentNodeSet you can create data connections to query XML data directly into your form.

Here are this module's contents:

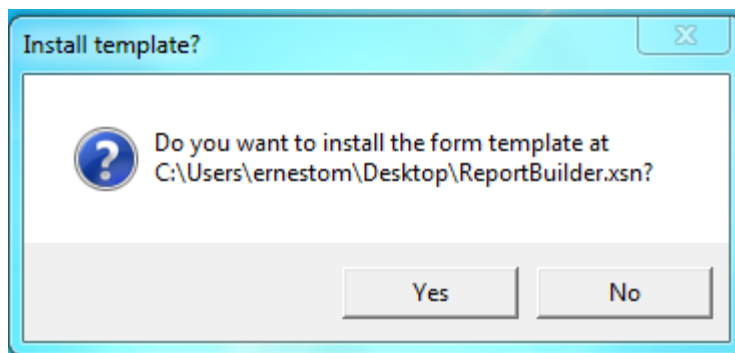
Install Report Builder	2
Create a new report	4
Update the existing report	8
Use QueryDocumentNodeSet in to avoid duplicates	9

INSTALL REPORT BUILDER

1. Use Internet Explorer to navigate to <http://servername/qdabrawebservice/>.
2. Click the first link, labeled **DBXL Administration Tool**. The DBXL Admin Tool (DAT) will open in InfoPath.
3. Click New Configuration and create a new document type called ReportBuilder_QDABRA. In the General tab, there's a checkbox for URN solutions. Please check this before saving.
4. Attach the Report Builder XSN and save the Document Type.
5. Download the XSN back into your local drive.
6. Make sure that InfoPath is not running (close all active InfoPath windows).

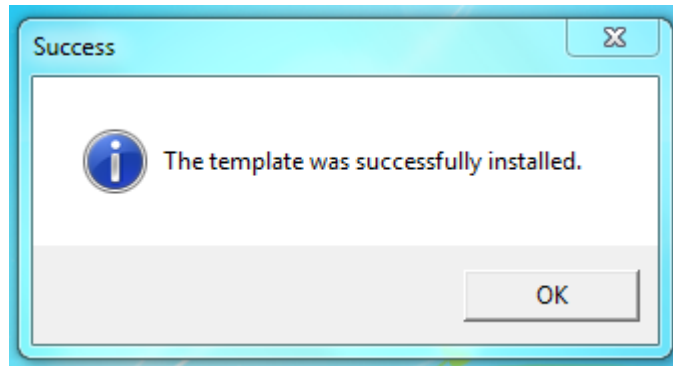
Note that sometimes, even after closing InfoPath, a process will remain active in the background. You can check this via the Task Manager.

7. Drag the **ReportBuilder** XSN (the one you downloaded from DAT) onto the **FormInstaller** icon. Click **Yes** when prompted.



8. **FormInstaller** will install the **ReportBuilder** form via a dialog. Click **OK** when done.

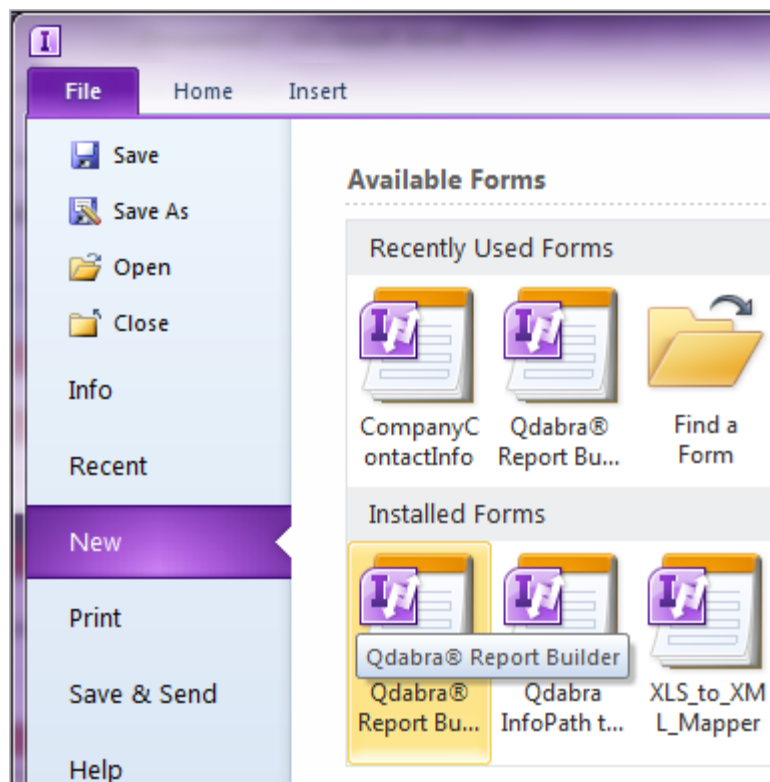
Qdabra DBXL Training: Module 6



CREATE A NEW REPORT

The following steps will show you how to create a brand new report.

9. Open InfoPath Filler.
10. Click **New**.
11. Select **Qdabra Report Builder** listed under the **Installed Forms** section.

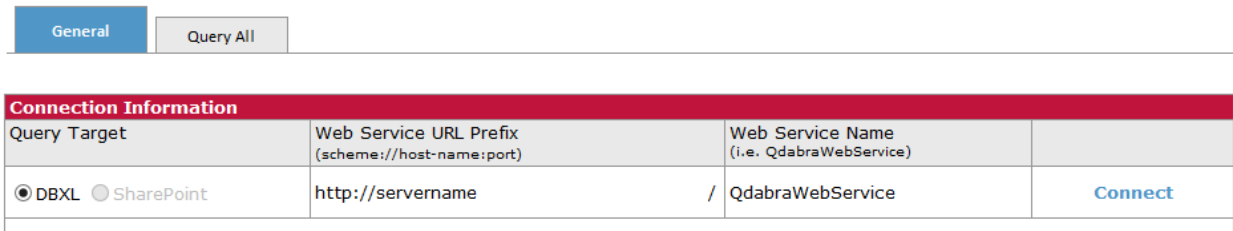


12. Under Report Name, provide a name or title for this report. If you'd like, provide a Description as well.



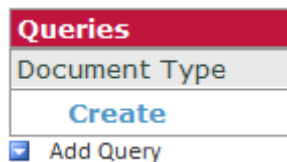
13. In the **General** section of the Report Builder tool, enter the Web Service URL Prefix.

14. Enter **QdabraWebService** as the Web Service Name and then click **Connect**.



Connection Information			
Query Target	Web Service URL Prefix (scheme://host-name:port)	Web Service Name (i.e. QdabraWebService)	
☒ DBXL ☐ SharePoint	http://servername	/ QdabraWebService	Connect

15. Click **Add Query > Create** in the *Queries* table.



16. Select the desired document type from the dropdown.

General	Query All
---------	-----------

Query Information	
Document Type	
ExpenseReportEventReceiver	▼

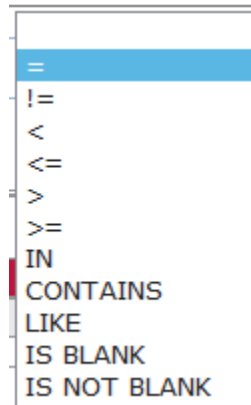
17. Under **Fields to Retrieve**, click on the Schema icon (). This will display the custom task pane at the right.
18. Double-click on the root node (for example, my:expenseReport).
19. Under **Field**, click **Insert Item** and start adding the fields you wish to retrieve. If desired, give each field an alias, which will become the label/title in the Excel report for that particular field/column. Note that we can't use spaces in aliases.

Fields to Retrieve	
Base Path	
/my:expenseReport	
Field	
my:employee/my:name	
my:manager/my:managerName	
my:items/my:total	
 Insert item	

20. If you have a large number of records in the document type, adding a filter is highly recommended. In the **Filter** section, make a selection in the dropdown:
 - **Simple Comparison:** For this type of filter, you'll compare ONE form field against ONE value.
 - **Complex Filter:** For this type of filter, you'll compare two or more fields.
21. Select **Simple Comparison** from the dropdown.

Qdabra DBXL Training: Module 6

22. Click on the Schema selector icon () to load the InfoPath form schema in the taskpane. Double click on the field you wish to use for your filter. In the middle dropdown select the desired comparison type. In the third column, enter the value for your filter.



23. Click **Perform Query** – you'll be taken to the *Query All* tab.
24. Click on the left arrow key () to expand and see the query result. If the results look OK, continue with the steps below. If you need to change the columns returned or the filter, click **Edit Query**, and return to previous steps to edit the query.

Returned Query Data			
Documents		Specify Export Workbook	Export All Queries to Excel
Document Type: ExpenseReportEventReceiver		Edit Query ▼	
docId	name	managerName	total
360	Ernesto		0
403	test	test	10
404	Ernesto Machado	Patrick Halstead	0
405	Ernesto Machado	Patrick Halstead	10
406	Ernesto Machado	Patrick Halstead	0

25. Click **Export All Queries to Excel**. In Excel save the spreadsheet to a desired location and with a desired name.
26. In Excel, you can create a **New Sheet** to perform calculations or use Pivot tables. However, you must leave the tab original sheet (created by Report Builder) intact.

UPDATE THE EXISTING REPORT

27. Close Excel and return to Report Builder.
28. Modify your query, for example, by adding a filter, or submit additional data to the form. Click **Perform Query** again.
29. Click **Specify Export Workbook**, and enter the full file path to the spreadsheet saved above.
30. Click **Insert Item**, and match the **Sheet Name** (on the left) to the **Query Name** (to the right). This allows Report Builder to update the data on the existing Excel sheet without altering any other sheets you may have added.

General

Query All

Returned Query Data

Documents

[Specify Export Workbook](#)
[Export All Queries to Excel](#)

Workbook Location:

C:\Users\ernestom\Desktop\test.xlsx

Sheet Name	Named Range	Query Name
ExpenseReportEventRe-1 ▼		ExpenseReportEventRe-1 ▼

31. Click **Export All Queries to Excel**. You'll notice that the original sheet will still be there, though refreshed with the data from the new filter. The sheet we added earlier is also still there, and updated with the new calculations.
32. When finished, click **Save Changes** to store this report in DBXL. You'll be able to come back to this report later on by clicking **Open** from the **Documents** tab.

USE QUERYDOCUMENTNODESET IN TO AVOID DUPLICATES

In this section we will add rules to our form to call QueryDocumentNodeSet and avoid duplicates.

- Using QueryDocumentNodeSet we can check the data that has been submitted to DBXL
- Ensure no duplicates are submitted
- Improves the user experience

33. Using Notepad, create a file called FormOptions.xml, whose contents are just:

```
<FormOptions>
    <IsNew>0</IsNew>
    <Exists>0</Exists>
</FormOptions>
```

34. Open the Expense Report form in Design mode.

35. In **Data > Data Connections**, click **Add**.

36. Create a new data connection that Receives data from an XML document. Select the FormOptions.xml file you created in the previous step, and finish the Data Connection Wizard, making sure that the connection executes on load.

We will need these nodes later on.

37. In **Data > Form Load**, find the Employee rule. Add an action that sets the **IsNew** node in the **FormOptions** secondary data source, to 1.

Qdabra DBXL Training: Module 6

Details for:

Employee

Condition:

emailAddress is blank

Rule type:

Action

Run these actions: *

Add ▾

Query using a data connection: GetMyInfo

Set a field's value: emailAddress = Value[Key = "mail"]

Set a field's value: name = Value[Key = "name"]

Set a field's value: employeeAlias = Value[Key = "sAMAccountNa...]

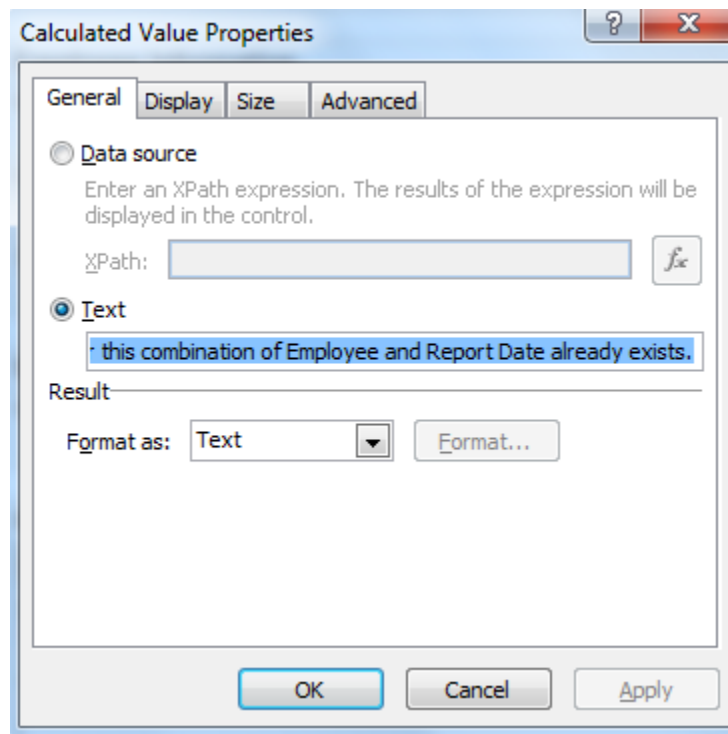
Set a field's value: IsNew = "1"

38. Add a Calculated value (Expression box) with the text: "A form for this combination of Employee and Report Date already exists."

EXPENSE REPORT

Report Date:	Expense Code:	Start Date:	End Date:
Business Purpose:			

A form for this combination of Employee and Report Date already exists.

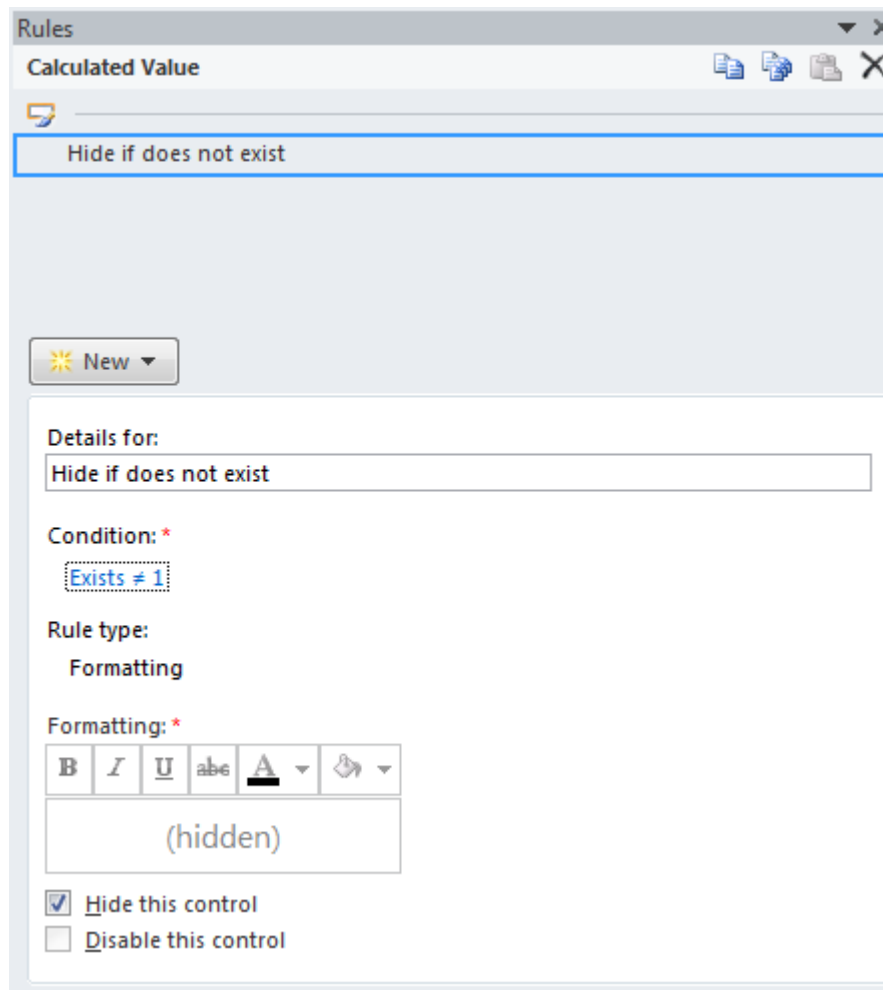


Now, for the Expression box, add a condition to hide it:

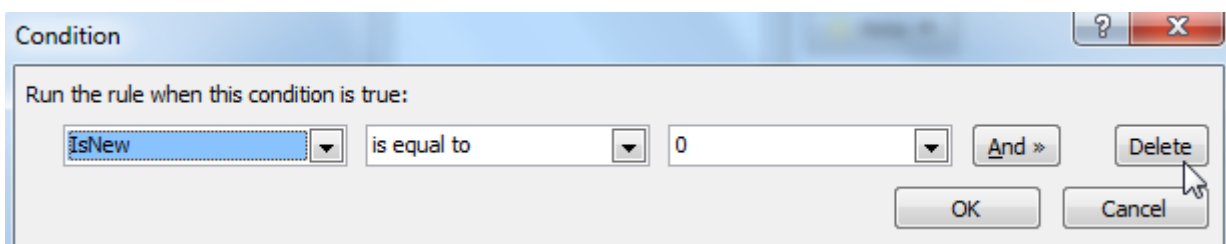
39. Select **New > Formatting**.

40. For the condition, you'll want to check the **Exists** field in the **FormOptions** secondary data source. If it is not equal to 1, we want to **Hide this control**.

Qdabra DBXL Training: Module 6



41. Add a second formatting, which will hide the Expression box and the link if the user has opened an existing record.



Before we proceed, we need to generate the queryXML via Report Builder.

Qdabra DBXL Training: Module 6

42. Launch the Report Builder by opening the InfoPath Filler and selecting Report Builder from the Installed Forms section.
43. Enter the Web Service URL Prefix, the Web Service Name, and then click **Connect**.
44. Under **Queries**, click **Add query**, then click **Create**.
45. Select the ExpenseReport document type from the dropdown.
46. Under **Fields to Retrieve**, click the selector icon ()
47. In the taskpane, double click the base node.

Fields to Retrieve
Base Path
/my:expenseReport

48. Click **Insert item**.
49. Click the selector icon () under **Field**.
50. In the taskpane, double click the node for the Employee's name.
51. Under **Filter**, change the dropdown for **Type** to **Complex Filter**.
52. Click **Insert item** twice.

This inserts two rows for filtering. Let's set the first one.

53. Click the selector icon () and double click the Employee name field in the taskpane.
54. In the middle dropdown, select the equal sign.
55. In the right most field, type in FIRSTPLACEHOLDER.

Now for the second filter.

56. Click the selector icon () and double click the reportDate field in the taskpane.
57. In the middle dropdown, select the equal sign.
58. In the right most field, type in SECONDPLACEHOLDER.

Qdabra DBXL Training: Module 6

The query should look like this:

General

Query All

Query Information

Document Type

ExpenseReportEventReceiver

Query Description

Fields to Retrieve

Base Path

/my:expenseReport

Field

my:employee/my:name

Alias

Insert item

Filter

Type

Complex Filter

AND

Insert logical sub-operation

/my:expenseReport/my:employee/my:name

=

FIRSTPLACEHOLDER

/my:expenseReport/my:reportDate

=

SECONDPLACEHOLDER

Insert item

You'll see that a query has been generated under QueryXML.

59. Copy the XML for query.

```
<query doctypeName="ExpenseReportEventReceiver"><columns
repeatingPath="/my:expenseReport"><column name="my:employee/my:name"
/></columns><filter><and><eq><column
name="/my:expenseReport/my:employee/my:name"
/><value>FIRSTPLACEHOLDER</value></eq><eq><column
name="/my:expenseReport/my:reportDate"
/><value>SECONDPLACEHOLDER</value></eq></and></filter></query>
```

Return to the form in design mode.

60. Go to **Data > Data connections** and click **Add**.

61. In the first screen of the Data Connection Wizard, select **Create a new data connection to Receive data**. Click **Next**.

62. Select **SOAP Web Service** and click **Next**.

63. Enter <http://<servername>/QdabraWebService/DBXLDocumentService.asmx> and click **Next**.

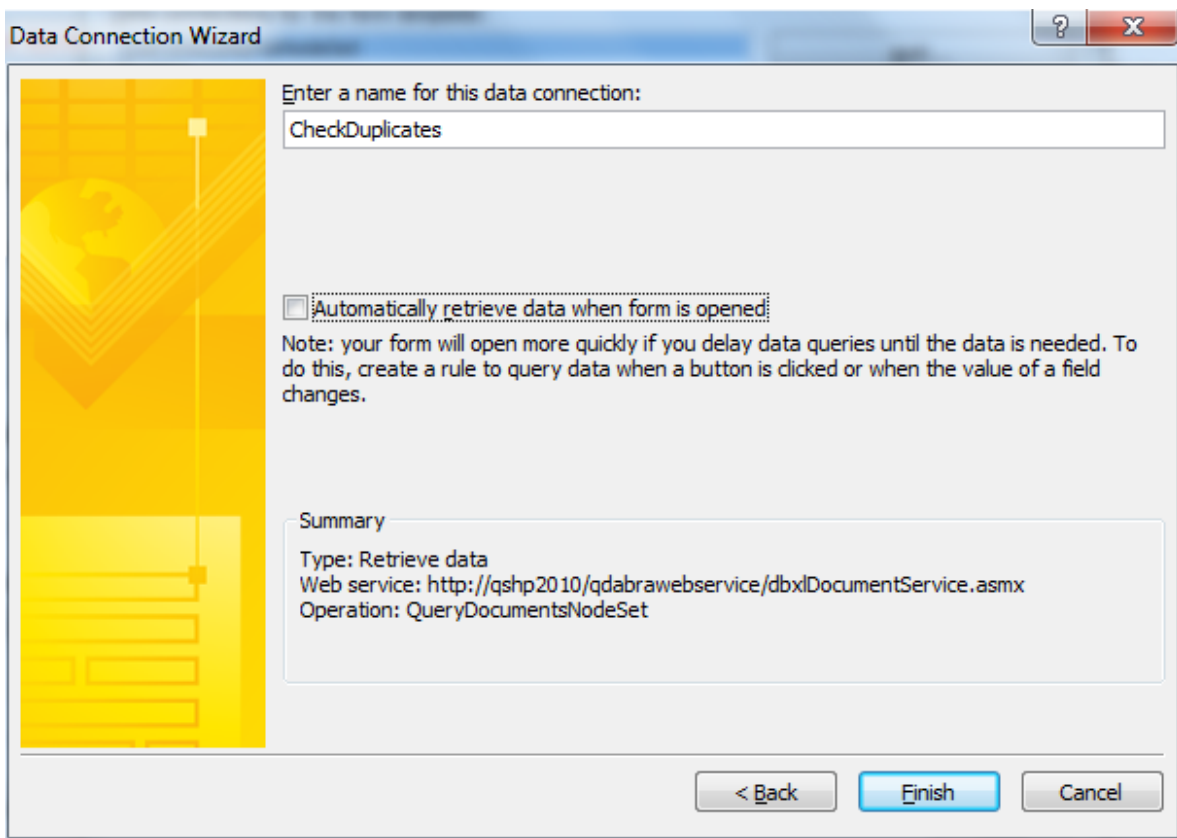
64. Select **QueryDocumentNodeSet** and click **Next**.

Now we need to use the query xml without the filter, for example:

```
<query doctypeName="ExpenseReportEventReceiver"><columns  
repeatingPath="/my:expenseReport"><column name="my:employee/my:name"  
</column></columns></query>
```

65. Enter the query, click OK and click **Next** twice.

66. Uncheck the option to retrieve the data on load, and click **Finish**.



The image shows a screenshot of the "Data Connection Wizard" dialog box. The title bar reads "Data Connection Wizard". On the left is a yellow decorative panel with a circuit-like pattern. The main area contains the following elements:

- A text field labeled "Enter a name for this data connection:" with the text "CheckDuplicates" entered.
- An unchecked checkbox labeled "Automatically retrieve data when form is opened".
- A note below the checkbox: "Note: your form will open more quickly if you delay data queries until the data is needed. To do this, create a rule to query data when a button is clicked or when the value of a field changes."
- A "Summary" section with the following text:
 - Type: Retrieve data
 - Web service: <http://qshp2010/qdabrawebservice/dbxlDocumentService.asmx>
 - Operation: QueryDocumentsNodeSet

At the bottom right are three buttons: "< Back", "Finish" (highlighted in blue), and "Cancel".

Qdabra DBXL Training: Module 6

67. Call this data connection “CheckDuplicates”. Click **Close** on the Data Connections screen.

Now that we have the data connection, let’s use it on load.

68. Return to **Data > Form Load**.

69. Click **New > Action**. Call this “Check duplicates”.

70. Add a condition such that this rule will execute when the employee **name** is not blank. If we are opening an existing record, then we don’t need to check this, so add a condition for that as well.

71. Click on **Add > Set a field’s value**.

72. For **Field**, select the query node from the **CheckDuplicates** data connection.

73. For **Value**, click **fx**, and enter:

```
concat('<query doctypeName="ExpenseReportEventReceiver"><columns
repeatingPath="/my:expenseReport"><column name="my:employee/my:name"
/></columns><filter><and><eq><column name="/my:expenseReport/my:employee/my:name"
```

Qdabra DBXL Training: Module 6

```

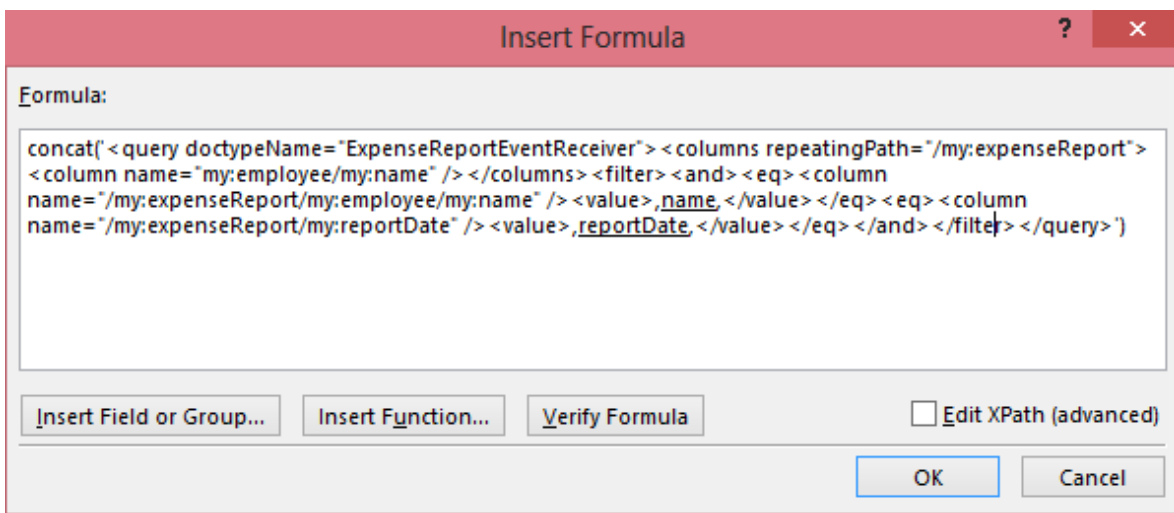
/><value>'FIRSTPLACEHOLDER'</value></eq><eq><column
name="/my:expenseReport/my:reportDate"
/><value>'SECONDPLACEHOLDER'</value></eq></and></filter></query>')

```

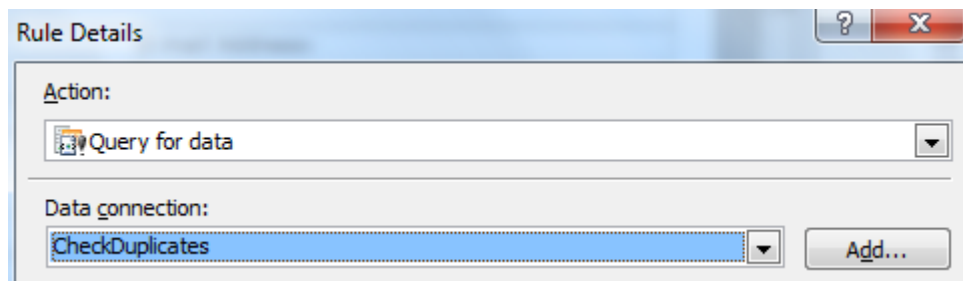
Where did this come from? This is the original query, generated by Report Builder, but we're using concat so that we can use the values in the form.

Notice the two locations where we have placeholder text.

74. Replace the first placeholder text with two commas. Then, in between the commas, click **Insert field or group**, so that you can enter the employee name and the ReportDate.



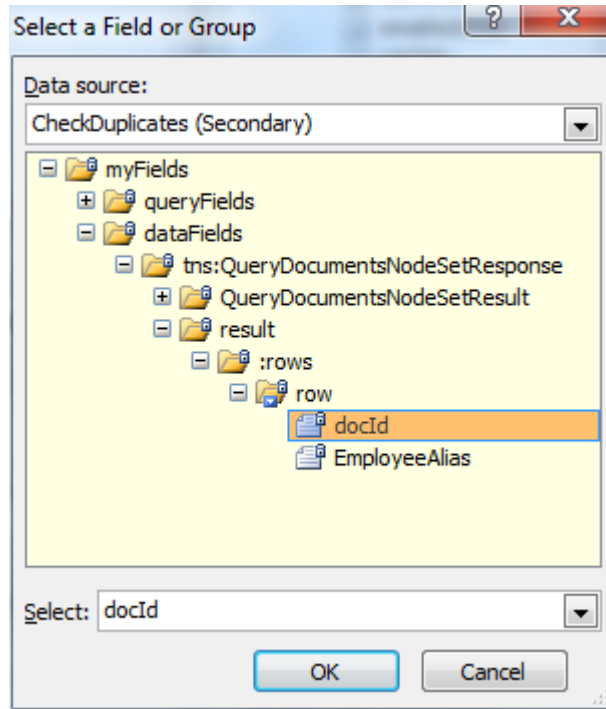
75. Click OK.
76. Click **Add > Query for data**, and select the QueryDocumentNodeSet data connection.



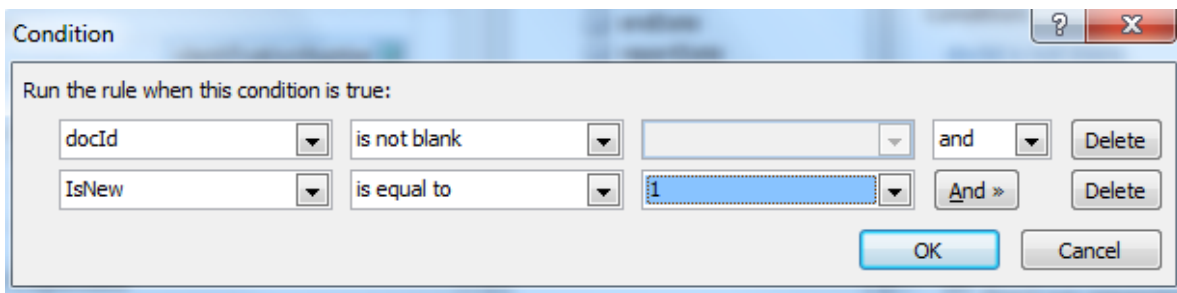
Finally, we need to set that flag in the secondary data source called FormOptions.

77. Click **New > Action**. Call this rule “Set flag”.

78. Add a condition for docId (from the CheckDuplicates results) is not blank.



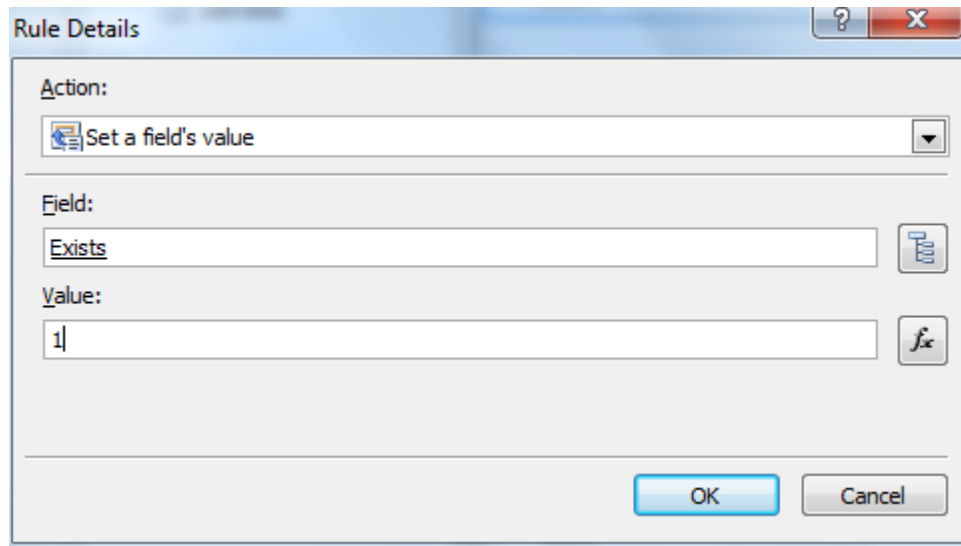
79. Again, we don’t need to check this if we’re opening an existing form, so add a condition for that as well.



80. Click on **Add > Set a field’s value**.

81. For the **Field**, select the **Exists** node from the **FormOptions** secondary data source.

82. For the **Value**, enter 1.



Rule Details

Action: Set a field's value

Field: Exists

Value: 1

OK Cancel

In other words, if a result was found, then a document **for this user on this report date** already exists!

83. Preview the form. Because you already submitted a document to DBXL, the form will detect this and hide the submit button, and display the message.



EXPENSE REPORT

Report Date: 3/24/2012 Expense Code: Start Date: End Date:

Business Purpose:

A form for this combination of Employee and Report Date already exists.

Employee Information

Name: Ernesto Machado E-mail Address: Ernesto.Machado@qdabra.com

Extra credit:

- Remember: if you want to use this method in the browser form, along with the Event Receiver, convert the data connection to UDC and use the SSO credentials.

Qdabra DBXL Training: Module 6

- Disable the submit button when showing the message (in red text, in the screenshot above).
- Add rules to the Report Date control, such that CheckDuplicates is called should the user change the ReportDate.