Product: Database Accelerator

Title: Implement Static and Dynamic Queries using QueryDB

In the QueryDB User Guide we discussed the possibilities offered by this web service. This document is a tutorial where we illustrate the use of Query Builder in conjunction with QueryDB to create two types of queries: static and dynamic.

Below you will find an outline of this document's contents.

Static query 2

Switch to a dynamic query 8

Hints and Tips 10

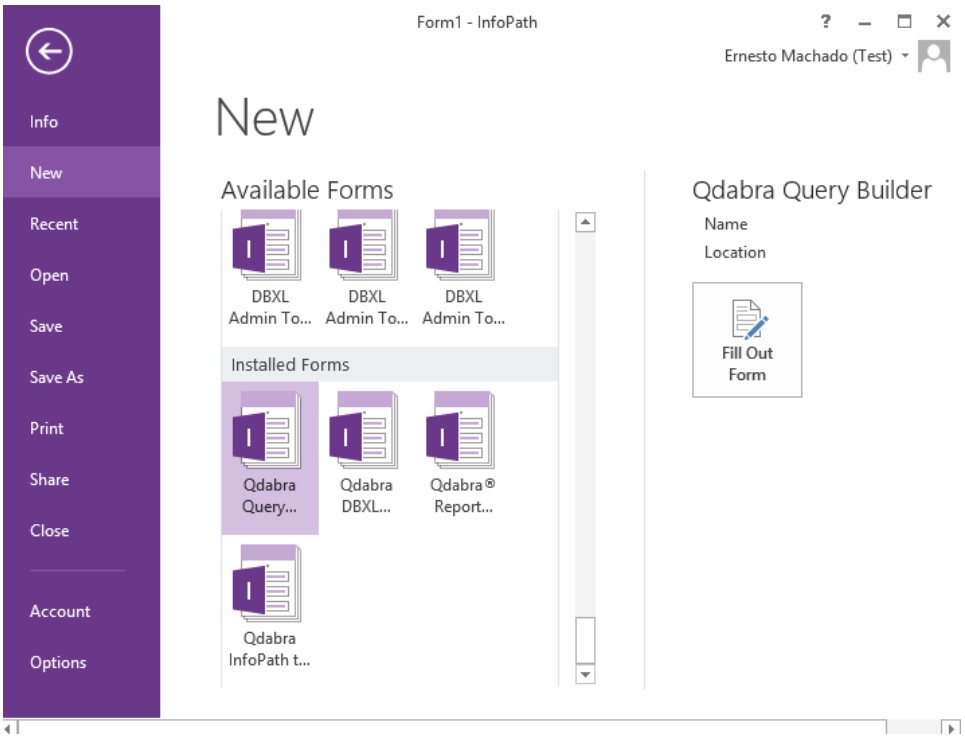
Support 12

STATIC QUERY

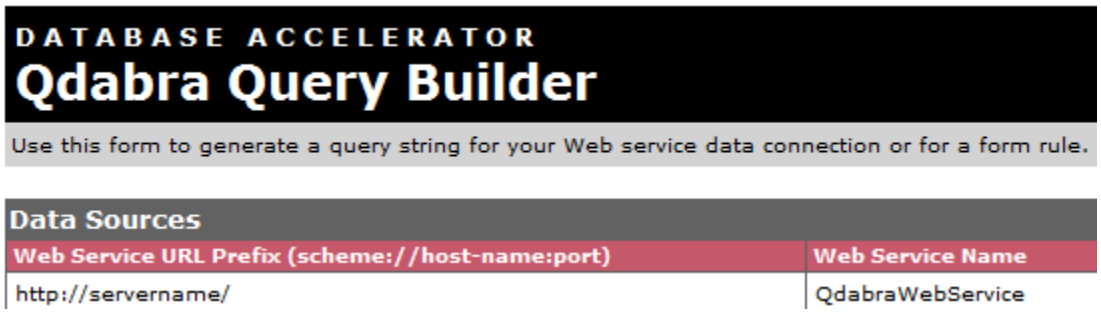
Before executing this tutorial, you must have installed Query Builder. For more information on installing Query Builder, please see: http://www.infopathdev.com/files/folders/other_subjects/entry81626.aspx

LAUNCH QUERY BUILDER

1. You'll find QueryBuilder in the Installed Forms section of the InfoPath Filler. Select it, then click **Fill Out Form**.



2. In the **Web Service URL Prefix** field, enter the URL to your host.
3. The **Web Service Name** is populated with QdabraWebService by default. You will only need to change this default value if you changed the value during installation of DBXL.



4. For the **InfoPath Form Template** field, you only need to attach an InfoPath form if you are going to be using a field from the form to build the query. We will do this in a future step, but for now this can be left blank.

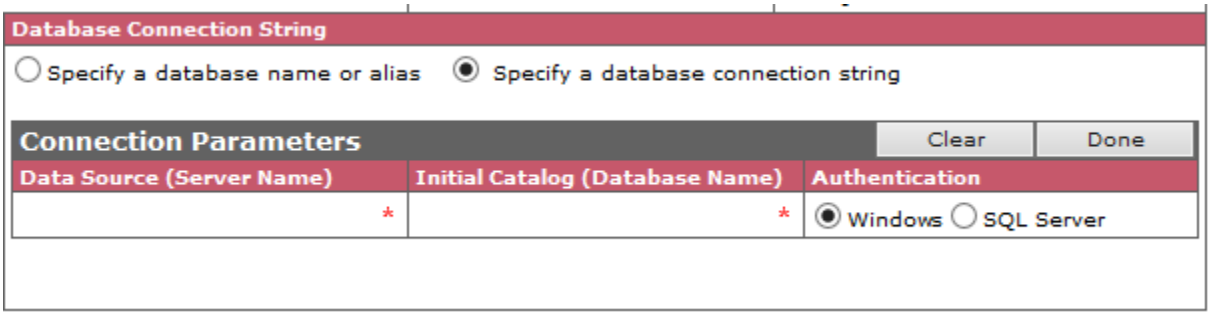
QUERYING A DATABASE

You will need to provide a database name or alias.

- If you are querying a database on the same SQL instance where DBXL was installed, uncheck "Is Alias" and enter the Database name.
- If you are querying a database on a different SQL instance, make sure **Is Alias** remains checked and enter the alias that was defined in the web.config. For more info, see the section below.

Then click **Refresh Database Schema**. This will populate the **Table Name** dropdown, where you can select the table you wish to query.

If you need Query Builder to help in building a connection string, click on "Specify a database connection string" and fill out the required fields before clicking **Done**.



In the **Query Parameters** section you can also specify the following, if desired:

- **Max Row Count**: The number of rows that will be returned by the query
- **Distinct**: Checkbox that tells the query to select only distinct values

ADDING A DATABASE ALIAS TO THE DBXL WEB.CONFIG

In order to allow QueryDB to access a database that lies on a different server, you must add the server and database aliases to the web.config file. Look for instructions within the configuration/qdabra.dbxl/enumdb/servers and configuration/qdabra.dbxl/enumdb/databases nodes. Note that the connection string in the <server> element in web.config should not specify a specific database, only the database server. The database alias should be separately added in the database aliases node as stated above.

```
<servers>
        <server name="default" connectionString="Data Source=(LOCAL);Integrated Security=True" />
        <server name="custom" connectionString="Data Source=REMOTE_SERVER;Integrated Security=True" />
</servers>
<databases>
        <database alias="QdabraSamples" name="QdabraSamples" server="default" />
        <database alias="DB_ALIAS" name="DATABASE_NAME" server="custom" />
</databases>
```

In the screenshot above we see the server named “default”, which is the SQL instance where DBXL was installed. In addition, a second server called "custom” has been added. You must provide a valid connection string in the servers entry. Then, in the databases entry, make sure to use the correct database name, and provide an alias. Then “Is Alias” is checked in Query Builder, you’ll see that Query Builder will automatically put pound (hash) signs around it, signifying that it’s an alias (and not a database name).

COMPLETE YOUR QUERY

The Query Columns section allows you to specify the columns that will be returned by your query. Whether you wish to query SQL or SharePoint, you’ll be able to select only the columns that need to be returned by the query.

5. Click on **Insert return column**.
6. Use the dropdown to select the column you wish to return.

Repeat to add as many columns as desired.

7. In the **Query Filter** table, click on **Insert comparison** to create a new query filter.
8. Use the dropdown to select the column to use in your filter. For our example, we will select **Client**.
9. Select a **Comparison Operator** from the dropdown. For our example, we will select **Equal**.
10. In the **Value or Column Name** field, enter *Client1*.
11. Click on **Insert Logical Operator**.
12. Select the logical operator **OR** from the dropdown.
13. Now repeat steps 8-10, replacing *Client1* with *Client2*.
14. In the **Query Sort** table, click on **Insert sort column**.
    a. Select a column from the dropdown, to use for sorting. For our example, we will select **Actual_length_months**.
    b. Select a **Sort Order** from the dropdown, in our case, we opt for **Ascending**.
15. Click on the **Build Now** button to generate the Query Strings. Copy the first two strings into Notepad. You’ll notice that the second and third string look identical. This is normal for static queries.

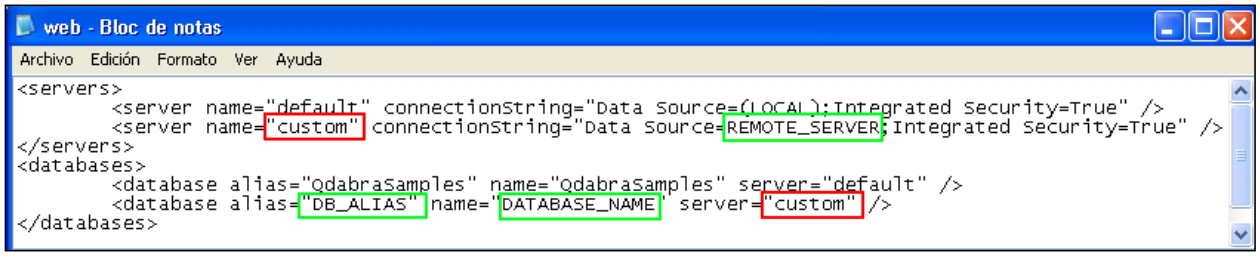
IMPLEMENT THE STATIC QUERY

16. Launch the InfoPath 2010 designer. Select blank form and click Design.
17. In the Data tab, click on **Data Connections**.
18. Click on **Add**.
19. Select **Create a New Connection to Receive data**. Click **Next**.
20. From the following screen, select **SOAP Web Service** and click **Next**.
21. Type in the URL for the QueryDB service, http://<yourservername>/QdabraWebService/QueryDB.asmx and click **Next**.
22. In the following screen, select **GetColumnsXMLStrQuery** and click **Next**.
23. Double-click on *tns:queryxml* and paste the first query that was generated by Query Builder. This is the sample value, used by InfoPath to derive a schema. Click **Next**.

You might receive an error at this point:

> ***The SOAP response indicates that an error occurred on the server:***
>
> ***System.Web.Services.Protocols.SoapException: Server was unable to process request. ---> Qdabra.QueryDB.QueryDbException: Error D0002:***
> ***----------***
> ***The database 'TestDB' is not registered for use***
> ***All databases used by QueryDB must be specified in web.config***

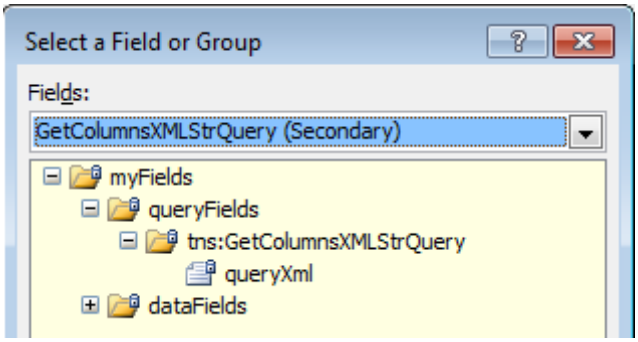
In order to allow QueryDB to access a database that lies on a different server, you must add the server and database aliases to the web.config file. Look for instructions within the configuration/qdabra.dbxl/enumdb/servers and configuration/qdabra.dbxl/enumdb/databases nodes. Note that the connection string in the <server> element in web.config should not specify a specific database, only the database server. The database alias should be separately added in the database aliases node as stated above.



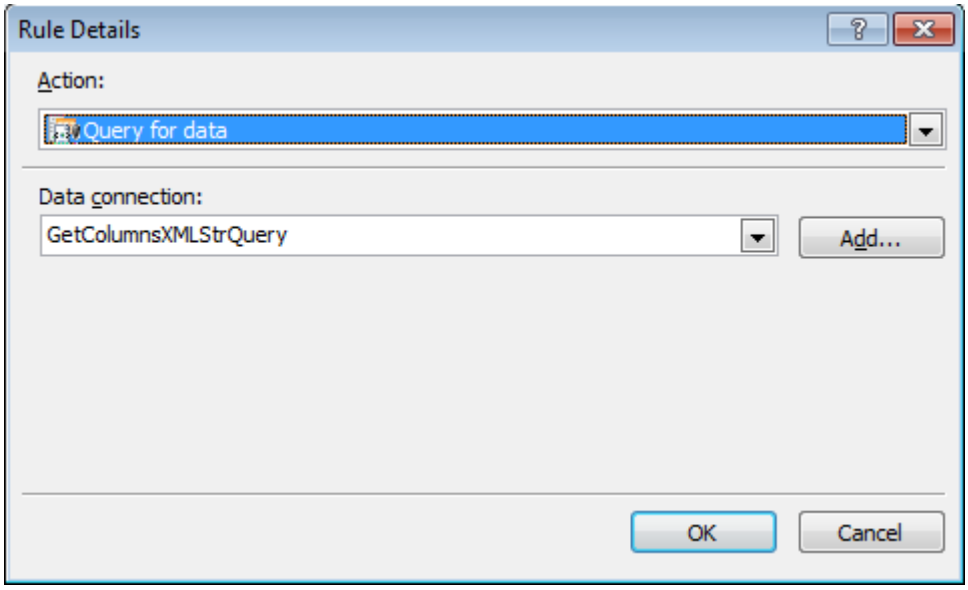
The database alias must be placed within pound signs (#, also called number sign or hash mark) when using QueryBuilder to build the queryxml and when adding the query to the InfoPath data connection wizard. Note that you must also provide the requester machine read access in the remote database.

This means that if the queryxml generated by QueryBuilder was ***<query database='TestDB' table='ClientInfo' maxrows='2' getSample='true'></query>***, you will need to enter ***<query database='#TestDB#' table='ClientInfo' maxrows='2' getSample='true'></query>*** into the data connection wizard.

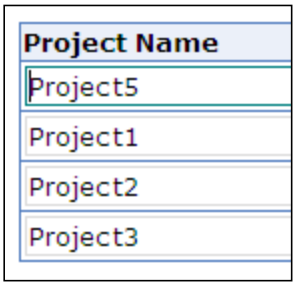
24. In the following screen, double-click on *tns:queryxml* and paste the second query that was generated by Query Builder. Remember to add pound signs (#) around the database alias if necessary. Click **Next** twice.
25. Ensure the checkbox for **Automatically retrieve data when this form is opened** is NOT checked. Ensure the name for this data connection is GetColumnsXMLStrQuery. Click **Finish**. Click **Close**.
26. In the Data tab, click on **Form Load**.
27. Add a new **Action** and name it **QueryDB**.
28. Next to "Run these actions" click **Add** and select **Set a field's value**.
29. For the **Field**, select the **queryxml** node in the GetColumnsXMLStrQuery secondary data connection.



30. For Value, enter the same queryxml you entered in step 9. Click **OK**.
31. Add a second action, which queries for data using the QueryDB data connection.



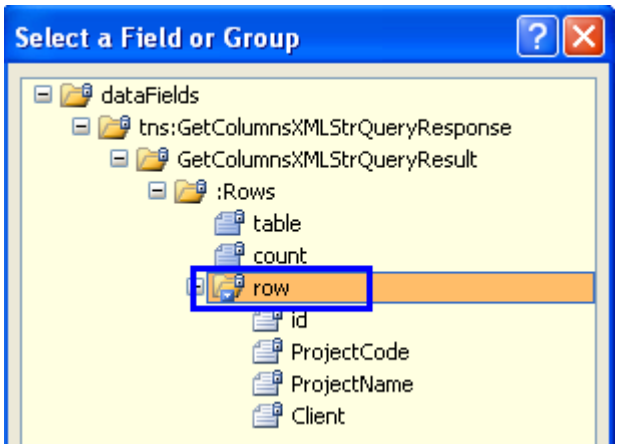
32. In the right hand pane called **Design Tasks**, click on **Data Source** and select GetColumnsXMLStrQuery from the dropdown.
33. Drag the node */dfs:myFields/dfs:dataFields/tns:GetColumnsXMLStrQueryResponse/tns:GetColumnsXMLStrQueryResult/Rows/row* and drop it into the form as a Repeating table.
34. Click on **File**, then **Save As**, and save the form.
35. Click on **Preview**. If prompted by a Microsoft InfoPath Security warning, click **Yes**. This will display the results, listing the projects for **Client1** OR **Client2**, sorted by the **Actual_length_month** column in ascending order.



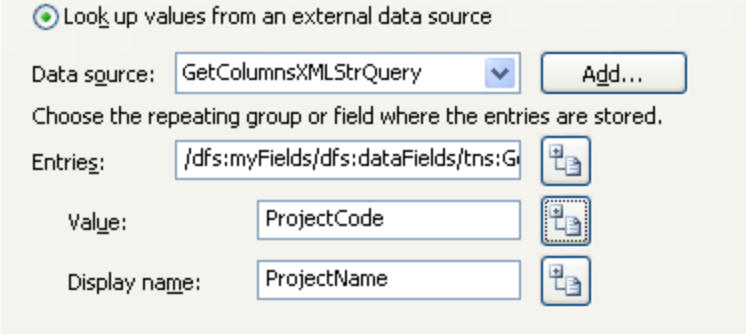
So far we have retrieved the data from SQL. Want to use it in a dropdown?

POPULATING A DROPDOWN

36. Add a dropdown list control on your canvas.
37. Configure the dropdown list properties
    - Right click on your dropdown list and select **Properties**.
    - Under **List box choices**, select "Get choices from an external data source".
    - Make sure the GetColumnsXMLStrQuery is selected for the **Data Source**.
    - Click **Select XPath** button for **Entries**, drill down to the lowest folder in the schema (row).



    - Select this node and click **OK**.
    - For **Value**, select the field you want to store in the XML.
    - For **Display**, select the field you want to display in your dropdown list. Click **OK** twice.



38. Test your form.
    - Preview your form. The form will show you the display and not the actual value. The value gets stored in the XML (and stored in SQL, if you set up a database mapping).

SWITCH TO A DYNAMIC QUERY

With a few additional steps we can implement a dynamic query that will call QueryDB after a user provides some missing piece of data. First we need to modify the form, then modify the query generated by Query Builder.

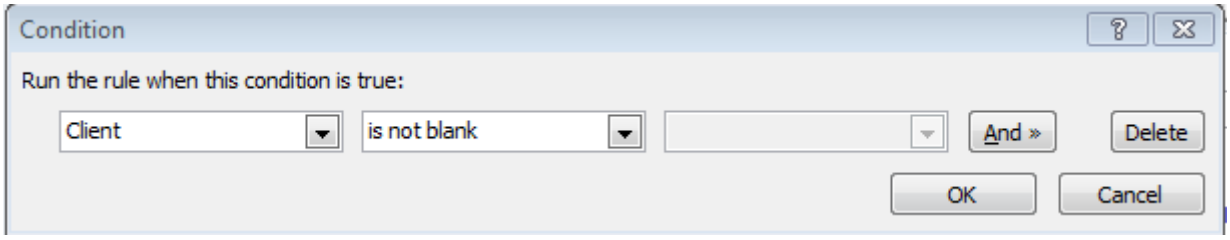
UPDATE THE FORM

39. Delete the repeating table you previously added to the form.
40. Add a Text box control to the view.
41. Right click on the textbox and select **Properties**. Change the field name to *client*. Click **OK** to close the textbox properties window. **Save** the form.

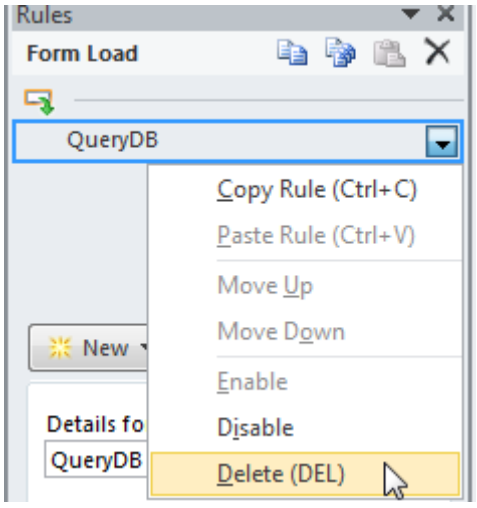
RETURN TO QUERY BUILDER

42. Launch Query Builder and fill out the Data Sources section. Refer to steps earlier in this document.
43. Attach your InfoPath form to the **InfoPath Form Template** file attachment control.
44. Click on **Insert comparison** and select the **Client** column for the **Column Name**. Leave the Comparison operator set to **Equal**.

45. For **Value or Column Name**, click on the **Select Schema Node** icon and select the *client* node in your InfoPath form. What this means is that the QueryDB data connection will only return the data that matches the value entered by the form user in the client node.
46. Click on **Build Now** and copy queries #1 and #3 into Notepad. (You will notice that the first and second queries are the same.)
47. Back in the InfoPath designer, select the Client text box and click **Manage Rules**.
48. Click **New** and select **Action**.
49. Add a condition which will execute this rule only when **Client** is not blank.



50. Under **Run these actions**, click **Add** and select **Set a field's value**.
51. Select the queryxml node from the GetColumnsXMLStrQuery secondary data connection. Then click **OK**.
52. Click on the icon to the right of the **Value** textbox and paste the third query generated by Query Builder. Remember to add pound signs (#), if necessary, to the database alias. Then click **OK** twice.
53. Click on **Add** and select **Query for data**.
54. Make sure that the data connection to GetColumnsXMLStrQuery is selected from the **Data connection** dropdown, and then click **OK**.
55. Under the **Data** tab, go to **Form Load** and remove the on-load rules we had previously added.



56. Save the form and click **Preview**.
57. Enter a client into the textbox and tab out of the field. Because the field is not blank, the rules will execute and the QueryDB data connection will gather the (filtered) data from the SQL table. The dropdown will show the projects associated with that specific client.

HINTS AND TIPS

INSTALLING QUERYBUILDER

Sometimes installing QueryBuilder will generate the following error: "***Cannot create InfoPath.ExternalApplication object. Error code = 0x80080005***." This error is usually avoided if you select the "Just Me" option when installing QueryBuilder (instead of "Everyone").

INVALID XML

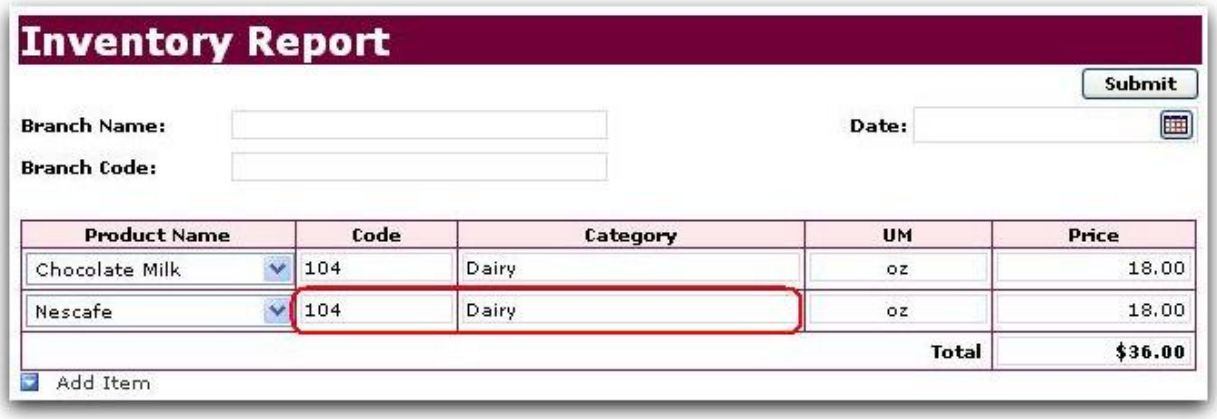
When working with QuerySharePoint or QueryDB, you might encounter an error that says: "Data at the root level is invalid. Line 1, position 1". This error indicates a problem with the <query> node. You could try using QueryBuilder to re-generate the value for <query>, but sometimes this indicates user error.

For example:

1. In a dynamic query, you should check that the <query> that begins with concat() is used in rules and not in the data connection wizard.
2. When entering the dynamic query into the rule, you must first click the function button (fx) before pasting the concat() query.

USING CURRENT() IN REPEATING TABLES

You might encounter a problem when using QuerySharePoint or QueryDB to dynamically retrieve values into a repeating table. To illustrate this scenario, imagine that a user selects a ***Product Name*** from a dropdown, and the goal is to set the ***Code*** and the rest of the repeating table fields with the corresponding values for the selected product. A second QuerySharePoint or QueryDB connection is created to dynamically query. The problem? The query (as generated by QueryBuilder) will not get updated when inserting new rows. The query always used the ***Product Name*** from the first row and returns the same value, as seen below.



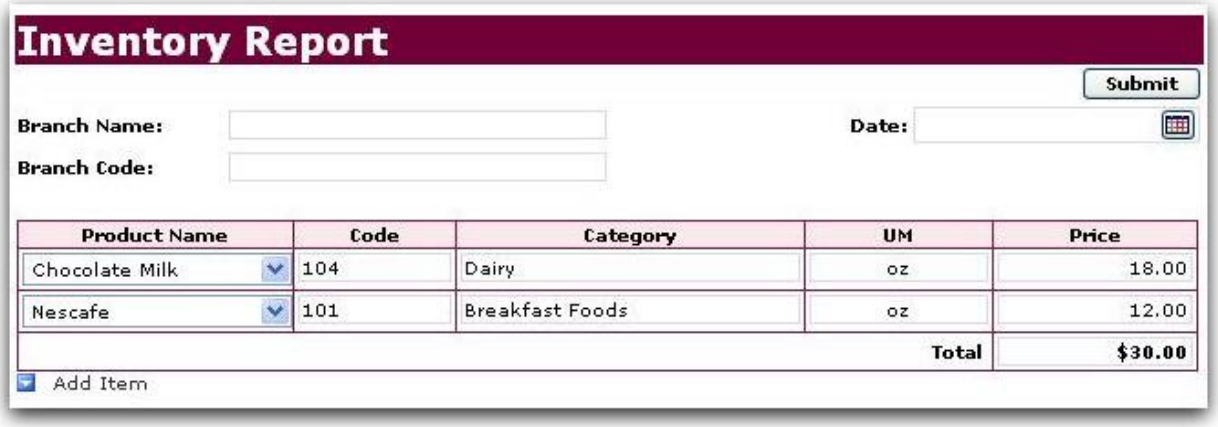
To identify the issue, we first need to look at the <query>that was created using Query Builder:

*concat("<query><columns><column name='Product_x0020_Code'/><column name='Category'/><column name='Unit_x0020_Of_x0020_Measure'/><column name='Price'/></columns><filter><eq><column name='Product_x0020_Name'/><value>", **ProductName**, "</value></eq></filter></query>")*

To fix this, simple replace ***ProductName*** with current():

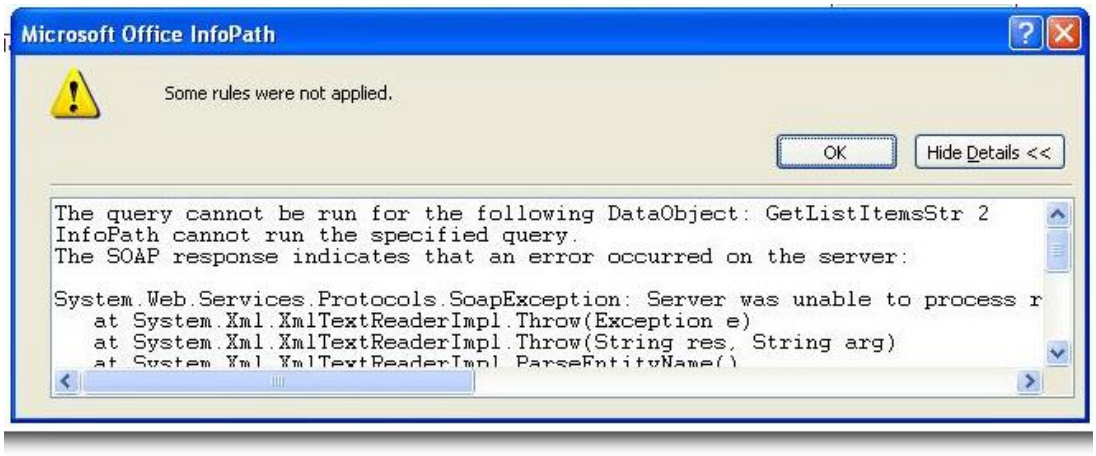
*concat("<query><columns><column name='Product_x0020_Code'/><column name='Category'/><column name='Unit_x0020_Of_x0020_Measure'/><column name='Price'/></columns><filter><eq><column name='Product_x0020_Name'/><value>", **current()**, "</value></eq></filter></query>")*

Now, when the user inserts a new row and selects a different ***Product Name***, we get the correct data:



USING CDATA FOR SPECIAL CHARACTERS

When SharePoint or database queries contain special characters, you might encounter an error such as the one seen in the screenshot below:



This means that our database or SharePoint data contains special characters, and we must edit the value of <query> to use CDATA.

concat("<query><columns><column name='Product_x0020_Code'/><column name='Category'/><column name='Unit_x0020_Of_x0020_Measure'/><column name='Price'/></columns><filter><eq><column name='Product_x0020_Name'/><value>", current(), "</value></eq></filter></query>")

What we need to do is just add a CDATA inside the query <value> </value>:

*concat("<query><columns><column name='Product_x0020_Code'/><column name='Category'/><column name='Unit_x0020_Of_x0020_Measure'/><column name='Price'/></columns><filter><eq><column name='Product_x0020_Name'/><value>**<![CDATA[**", current(), "**]]>**</value></eq></filter></query>")*

Note that CDATA should always start with the sequence: **<![CDATA[** and ends with **]]>**.

USING COUNTONLY

The countOnly parameter allows you to return only the number of items, instead of returning the entire data set. To set the parameter to true, use a query like this:

*<query **countOnly='true'** database="MyDB" table="MyTable"></query>*

If the countOnly paremeter is equal to anything other than 'true' (or if it is not present), then it is considered to be equal to false, and all records are returned.

SUPPORT

If you have questions about the information in this document, please contact us for assistance.

Licensed customers contact us via **Support@Qdabra.com**. Alternatively, please use the InfoPathDev.com Qdabra Product support forums to request help.