



Product: Database Accelerator (DBXL) v2.5

Title: QuerySharePoint User Manual

This is the User Manual for the QuerySharePoint web service. Below you will find an outline of this document's contents.

Operations	2
DescribeListSchema and DescribeListSchemaSiteAndGuid	2
GetDropdownEntries and GetDropdownEntriesSiteAndGuid	2
GetListItems and GetListItemsSiteAndGuid	2
GetListItemsStr and GetListItemsStrSiteAndGuid	2
Parameters	3
Filtering	3
Hints and Tips	4
Installing QueryBuilder	4
Invalid XML	4
Using Current() in Repeating tables	4
Using CDATA for special characters	6
QueryBuilder does not show user-friendly column names	7
QuerySharepoint cannot derive a schema with blank fields	7
Using countOnly	8
Encoded spaces in the URL produce an error	8
Support	8



OPERATIONS

DESCRIBELISTSHEMA AND DESCRIBELISTSCHMASITEANDGUID

DescribeListSchema scans the schema for the specified SharePoint list and returns an XML Document describing it. The **listUrl** parameter should be the root URL of a SharePoint list.

Use DescribeListSchemaSiteAndGuid (instead of DescribeListSchema) if the list may be moved or renamed in the future since the list name guid is less likely to change. The **listId** should be the guid uniquely identifying the SharePoint list on the site.

GETDROPDOWNENTRIES AND GETDROPDOWNENTRIESSITEANDGUID

GetDropdownEntries allows a user to retrieve a list of values and easily bind them to a dropdown control. By default, the values will be sorted according to valueColumn in ascending order, or by displayColumn if provided. The user can also indicate sortOrder (ascending or descending) and a filterXML. Here is the full information regarding the parameters:

- **listUrl**: the URL to the SharePoint list
- **valueColumn**: the column name for default sorting
- **displayColumn** (optional): column to populate display field in results
- **sortColumn** (optional): can be any column in the table
- **sortOrder** (optional): "ASC" or "DESC" for ascending or descending
- **filterXml** (optional): a <filter> xml, generated in Qdabra's Query Builder form.

Use GetDropdownEntriesSiteAndGuid (instead of GetDropdownEntries) if the list may be moved or renamed in the future, since the list name guid is less likely to change. Instead of providing the listUrl, indicate the siteUrl and the listId.

GETLISTITEMS AND GETLISTITEMSSITEANDGUID

This operation performs GetListItems on the SharePoint list and returns the resulting rowset. The listUrl is the root URL of the SharePoint list.

Use GetListItemsSiteAndGuid (instead of GetListItems) if the list may be moved or renamed in the future, since the list name guid is less likely to change. Instead of providing the listUrl, indicate the siteUrl and the listId.

GETLISTITEMSSTR AND GETLISTITEMSSTRSITEANDGUID

GetListItemsStr is similar to GetListItems, except that the arguments, <query> and <queryOptions>, are strings (instead of xml).



Similarly, `GetListItemsStrSiteAndGuid` is analogous to `GetListItemsSiteAndGuid`, except that the arguments, `<query>` and `<queryOptions>`, are strings (instead of xml).

PARAMETERS

- **listUrl** is the URL of a SharePoint library which can be a document, form or list library. Examples:
 - `http://<SharePointSite>/<SharePointLibrary>/Forms/AllItems.aspx`
 - `http://<SharePointSite>/<SharePointLibrary>/`
 - `http://<SharePointSite>/<SharePointLibrary>`
- Note:** To query SharePoint lists, make sure not to add `Forms/AllItems.aspx` in the `listUrl`. Example:
 - `http://<SharePointSite>/Lists/<SharePointList>/`
 - `http://<SharePointSite>/Lists/<SharePointList>`
- **siteUrl** is the root Url of the SharePoint site. Examples:
 - `http://<SharePointSite>`
 - `http://<SharePointSite>/sites/<SharePointSubsite>`
- **listId** should be the guid uniquely identifying the SharePoint library. It should contain 32 digits with 4 dashes (xxxxxxxx-xxxx-xxxx-xxxxxxxxxxxx) and can be obtained through `DescribeListSchema` operation.
- **webId** is the guid that uniquely identifies the site (web) in a SharePoint site collection. WSS 3.0 requires this in the API – `listId` is not enough. It should contain 32 digits with 4 dashes (xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx) and can be obtained through `DescribeListSchema` operation.
- **query** is a filtering operation that is generated through Qdabra's Query Builder form.
- **queryOptions** is based on SharePoint's schema of the same name.

FILTERING

`GetDropdownEntriesSiteAndGuid` and `GetDropdownEntries` allow you to create a filter, which filtering queries can be easily built using Qdabra's Query Builder tool.

- a. **lt, le, eq, ne, ge, gt** - The first set of comparison operators all use similar formats. The comparisons can take place between a column and a value, or between a column and another column.
 - `<eq>`: "equal"
 - `<ne>`: "not equal"
 - `<gt>`: "greater than"
 - `<ge>`: "greater than or equal"
 - `<lt>`: "less than"
 - `<le>`: "less than or equal"
- b. **in** - The use of `<in>` is slightly different from the above set of comparison operators. This operator requires the user to specify a column and at least one value. It then selects rows that contain the value(s) listed in the specified column. The following example illustrates this:

`<in>`



```

        <column name="Client"/>
        <values>
            <value>Client1</value>
            <value>Client2</value>
            <value>Client3</value>
        </values>
    </in>

```

- c. **NULL** - One useful and common filtering operation involves the search for NULL values in your database. In order to do this, we will need to use `<ne>` and `<eq>`. NULL values can be excluded, as shown below:

```

    <ne>
        <column name="Client"/>
        <value nil="true"/>
    </ne>

```

NULL values can also be included:

```

    <eq>
        <column name="Client"/>
        <value nil="true"/>
    </eq>

```

- d. **CONTAINS** - The `<contains>` filter allows you to perform keyword searches in your database, in the same way you should use it in a SQL statement (see <http://msdn2.microsoft.com/en-us/library/ms187787.aspx>). Note that you cannot use `<contains>` in conjunction with a NOT logical operator.

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>`.

For example, 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.

USING CURRENT() IN REPEATING TABLES



<http://www.qdabra.com>

Last updated on 4/1/2011 11:40 AM

Copyright © 2006-2010 Autonomy Systems, LLC. All rights reserved.

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.

Product Name	Code	Category	UM	Price
Chocolate Milk	104	Dairy	oz	18.00
Nescafe	104	Dairy	oz	18.00
Total				\$36.00

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, simply 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:



Inventory Report

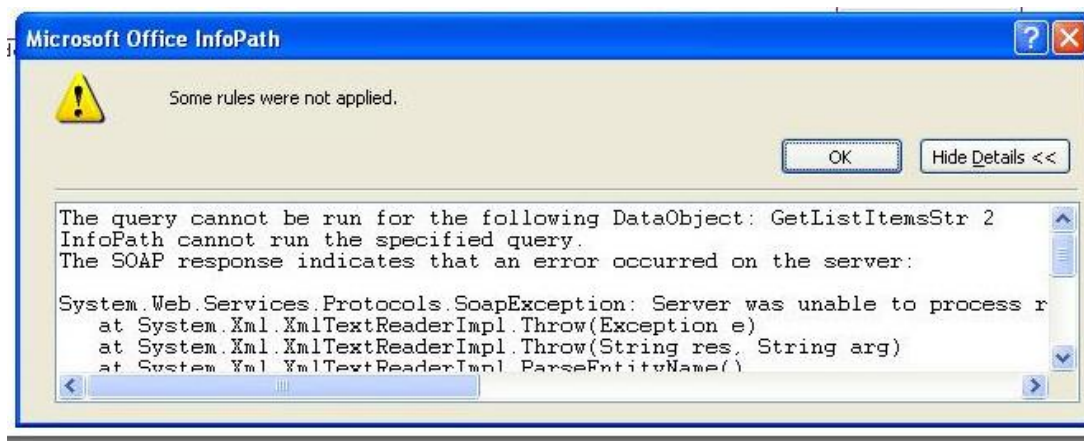
Branch Name: Date:

Branch Code:

Product Name	Code	Category	UM	Price
Chocolate Milk v	104	Dairy	oz	18.00
Nescafe v	101	Breakfast Foods	oz	12.00
Total				\$30.00

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]]>.



QUERYBUILDER DOES NOT SHOW USER-FRIENDLY COLUMN NAMES

When using QueryBuilder to build SharePoint queries, you might notice that QueryBuilder might display column names that are not user-friendly, as in the screenshot below.

Qdabra Query Builder

Use this form to generate a query string for your Web service data connection or for a form rule.

Data Sources		
Web Service URL Prefix (scheme://host-name:port)	Web Service Name	Target
http://dbxl23demo	QdabraWebService	SQL Database ☐ SharePoint List ☒
InfoPath Form Template (XSN)	SharePoint List URL	
Click here to attach a file	http://dbxl23demo/Test%20Lib/Forms/AllItems.aspx	

Query Parameters	
Parameter Name	Max Row Count

Query Columns	
Column Name	
_x007b_857c8175_x002d_705a_x002d_460a_x002d_aee3_x002d_91721bc3bf71_x007d_	
_x007b_0ca90904_x002d_6f51_x002d_4c97_x002d_a451_x002d_b908a75acbc5_x007d_	

Field Selection

SharePoint List

- Test Lib
 - Check In Cor
 - Copy Source
 - Version
 - Form Name
 - Sample Field
 - Created By
 - Checked Out
 - Content Type
 - Created
 - Type
 - Edit
 - Modified By
 - Name
 - File Size
 - ID
 - Name
 - Name
 - Modified
 - Source Name
 - Source Versic

The **<query>** generated by QueryBuilder will still work, in spite of the column names.

QUERYSHAREPOINT CANNOT DERIVE A SCHEMA WITH BLANK FIELDS

QueryBuilder allows you to specify the columns you wish to retrieve.

Query Columns

Column Name
Category
Unit_x0020_of_x0020_Measure
ProductCode

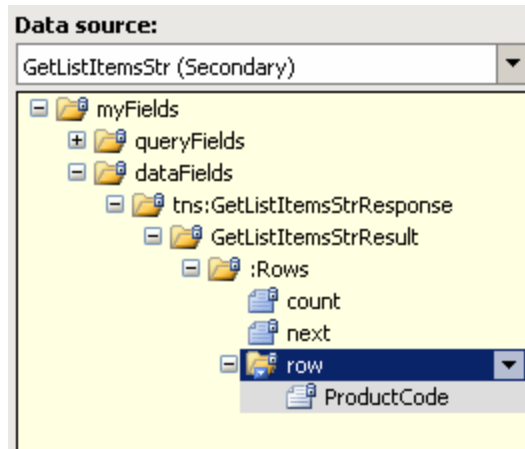
[Insert entire column](#)

The screenshot above shows three columns being queried by QueryBuilder. The **<query>** that is generated looks like this:

```
<query maxrows='2'><columns><column name='Category'></column
name='Unit_x0020_of_x0020_Measure'></column name='ProductCode'></columns></query>
```

However, the data connection may not return all three columns, as seen below:





This happens when the source data contains blank values, which means that the web service was unable to derive the correct schema. The workaround is simple: remove the maxrows parameter:

```
<query><columns><column name='Category'/><column
name='Unit_x0020_of_x0020_Measure'/><column name='ProductCode'/></columns></query>
```

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 parameter 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.

ENCODED SPACES IN THE URL PRODUCE AN ERROR

When using QueryBuilder and QuerySharePoint, you need to enter the URL to your SharePoint list. Internet Explorer encodes spaces as %20, but DBXL produces an error if you use a URL containing a %20.

The workaround is simply to replace %20 in the URL with an actual space. This works in QueryBuilder as well as in the InfoPath data connection wizard.

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](http://Qdabra.com) [Product support forums](http://ProductSupport.com) to request help.



<http://www.qdabra.com>

Last updated on 4/1/2011 11:40 AM

Copyright © 2006-2010 Autonomy Systems, LLC. All rights reserved.