

Knowledge Base Article: Article 255

**How to use the new Click&DECiDE Queries Data Source?
(Allowing queries on various data sources at the same time)****Last Update:** February, 16th, 2017**Requirement:**

You need to install Click&DECiDE Version 16 (64-bit) or greater.

Introduction:

Click&DECiDE version 16 provides a new data source named “**Click and DECiDE Queries**” allowing to use the queries from a Click&DECiDE project file (*.wfv) as a Data Source. If that project contains several queries made against various data sources, you can create a new query using each existing query as if it was a table. You can thus join data coming from different data sources. For example, one query has been made against an iSeries data source, another query has been made against an Oracle data source, another one against an SQL Server data source or against an Excel data source.

Targets:

This new data source has two targets:

1. To be able to make queries against various data sources at the same with the possibility to define some caches, scopes or index. The query result is stored in some hidden tables.
2. To be able to create a data warehouse using the Click&DECiDE automatic ETL feature whose target is to prepare for end-users some « synonyms », that means some views pointing to hidden tables defined in point 1.

1- Direct « Click and DECiDE Queries » data source:

Cette partie est plutôt réservée à des utilisateurs avertis ou aux administrateurs.

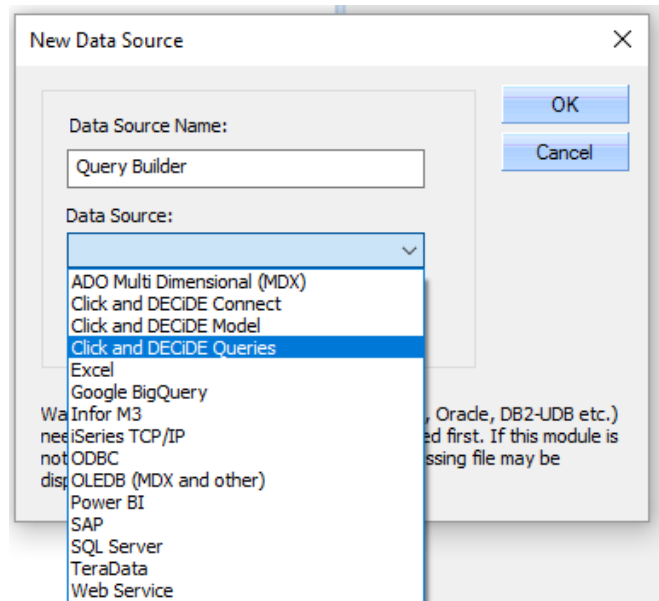
- The first step consists of using the new « Click and DECiDE Queries » data source to define some caches for the result (in various hidden tables) being updated according to the cache parameters, the « scopes » option (result set in cache by user or Globally) and with optional indexes.
- The second step consists of using this new « Click and DECiDE Queries » data source to join X queries from different data sources. By default, if « caches » or « scopes » or « index » are not defined, the result for each will be set in cache for 5 minutes by user.

2- Create and use synonyms with « Click and DECiDE Queries » data source:

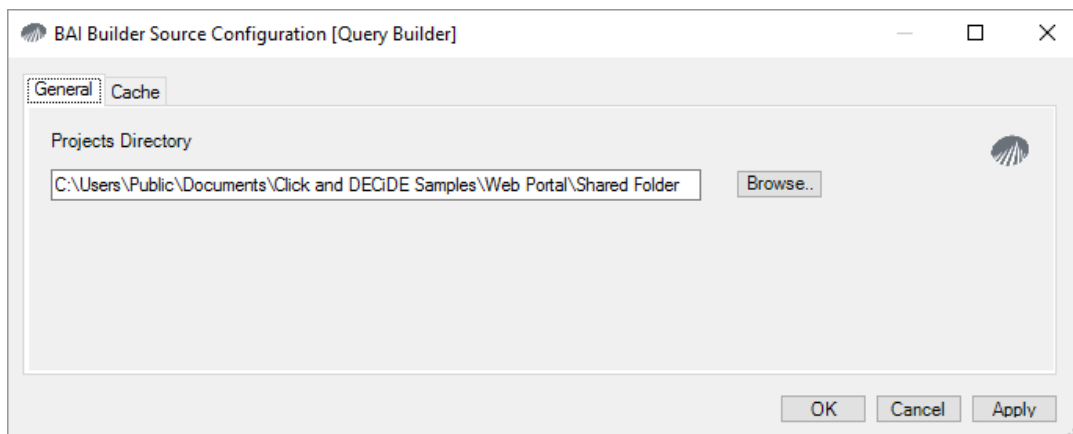
- In the above target point 2 some advanced users or administrators can use this new « Click and DECiDE Queries » data source to create some synonyms, from single or multi data source queries.
- The synonyms are « views » pointing to the result saved with a cache for each table used in the query.
- The benefit to use synonyms is to give a unique name to this result, easy to understand by the end-users.
- The end-users will use a standard SQL Server data source to access to these synonyms.
- Note that in the synonyms, some of them can be created from within an iSeries, other from Excel or SQL Server etc., and that the end-user working with these synonyms can create queries against various data sources without knowing that, because all synonyms are made under SQL Server.

Use:**1- Create the Click and DECiDE Queries data source:**

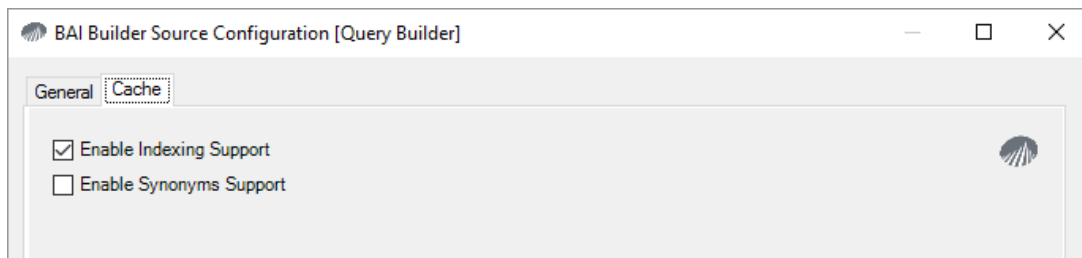
1. Open Click&DECiDE Administration Manager and select the Data Source Tab.
2. Make a right mouse click and select New Data Source.
3. Enter a name and select the Click and DECiDE Queries data source in the list box:



4. Validate with OK
5. Specify the working directory where Builder can find the appropriate project files (*.wfv), for example a shared folder from the Web Portal or a directory in the server not visible through the Web Portal):
Example: C:\CnD\My projects or Builder or
C:\Users\Public\Documents\Click and DECiDE Samples\Web Portal\Shared Folder)



6. In the Cache Tab, you can specify if you wish to enable some options:

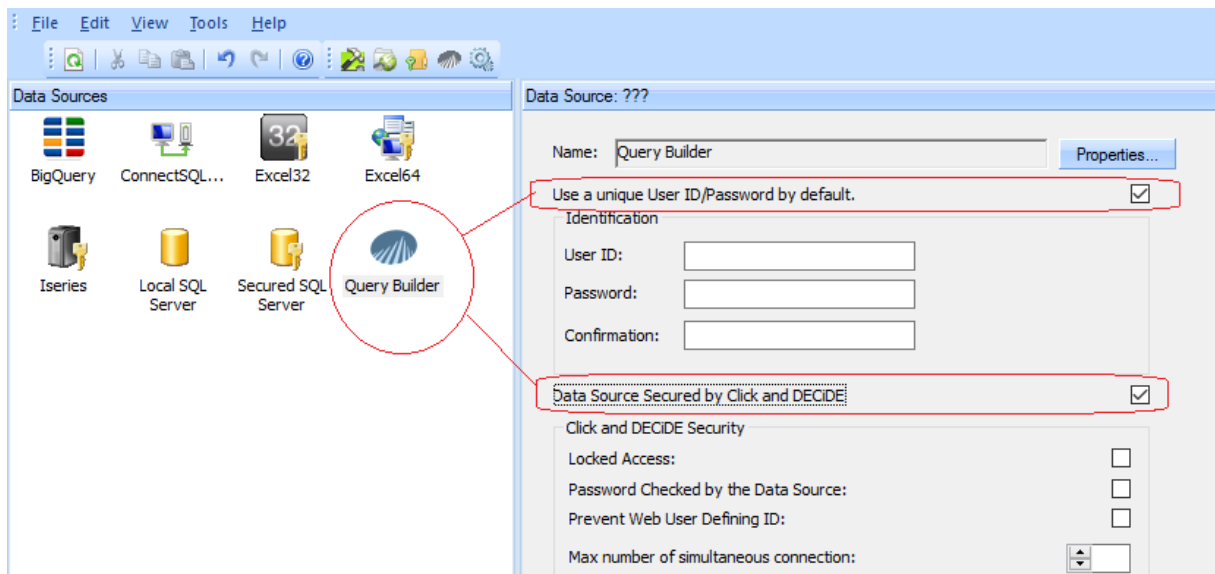


Enable Indexing Support: means that you want to be able to define some indexes for some fields.

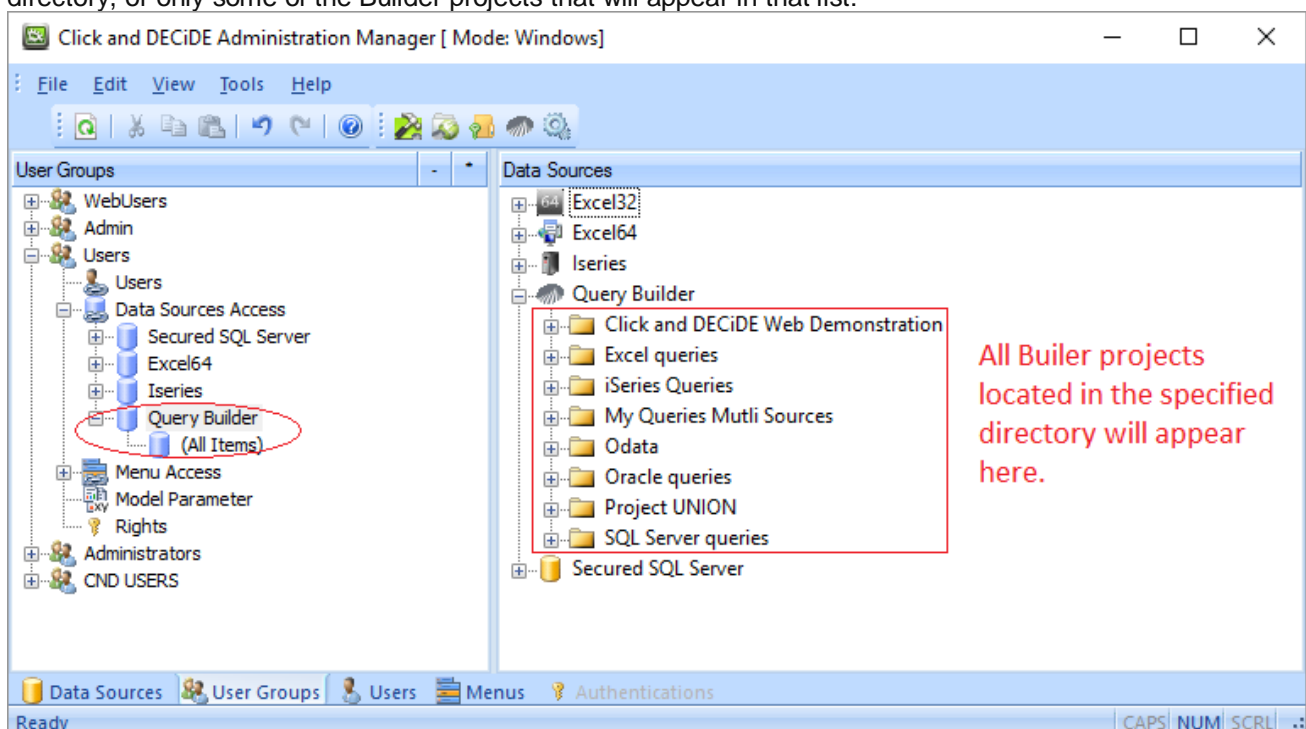
Enable Synonyms Support: means that you want to be able to have access later to an end-user table corresponding to the result of the query you have created. That option will be described in the second part of this manual 2- *Create and use synonyms with the « Click and DECiDE Queries » data source.*



7. Validate with OK. Make a right mouse click on this new “Query Builder” data source and click “Test connection”.
8. If you are using Windows Authentication, just click OK.
9. If you are using Click&DECiDE Authentication, enter your UserID and password then click OK.
10. If this data source has to be used through the Web Portal, define it as Secured by Click&DECiDE:



11. Activate “Use a unique User ID/Password by default », without the need to enter a login if under Windows Authentication.
12. Add this secured data source the concerned User Groups.
13. You can add the full data source, that means automatically all projects being or that will be in the working directory, or only some of the Builder projects that will appear in that list:



14. Click **File > Update Security** from Click&DECiDE Administration Manager.
15. Note that the Click and DECiDE Queries data source can access several queries saved in a same project or saved in distinct project files.



Use from Click&DECiDE Builder

➤ Case of a query source having no parameters.

1. Start Builder (or Web Builder from the Web Portal)
2. Open or create a project (for example My Queries Multi Sources.wfv) file.
3. If the "Query Builder" data source has been enabled to the User Group you belong to, you should see the authorized project files available, or for all Groups you belong to.
4. Create a new query against this « Query Builder » data source:
Select the required project to display the existing queries, as table-sources, for example the project containing Excel queries:

Name	Description
Sales	Sales
sales_details	Detailed sales
salesman	Salesmen

Name	Type	Length	Scale	Description
SAL	Float	53	0	Salesman code
SALNAME	Varchar	255		Salesman name
MAIL	Varchar	255		Salesman e-mail
AREA	Varchar	255		Salesman area

5. Note that the Description only appears on the right side if you have entered a text in the "Description" box in the Properties General Tab for the concerned query in the Builder project:

Name	Data Source	Description
Sales	Excel64	Sales
sales_details	Excel64	Detailed sales
salesman	Excel64	Salesmen

Properties	
General	
Name	salesman
Data Source	Excel64
Description	Salesmen
Created	01/17/2017 1:53:24 PM
Modified	01/19/2017 4:08:57 PM
Author	Didier

6. Select a table from this data source, for example "Salesman" from the "Excel queries" data source. Make a drag and drop this table. Then select another data source and another table from this other data source, for example "Sales" from the "SQL Server queries" data source:



The screenshot shows the 'All Data Sources' pane on the left with 'SQL Server queries' highlighted. The 'Content of [Query Builder]: "SQL Server queries"' pane on the right lists several tables: Customer, Product-Family-Category, sales, sales_details, Salesman, salesman list, and state. The 'sales' table is highlighted with a red box, and a red arrow points from it to the 'sales' table structure shown below. The 'salesman' table structure is also shown.

Name	Type	Length	Scale	Description
SAL	Float	53	0	Salesman code
SALNAME	Varchar	255		Salesman name
MAIL	Varchar	255		Salesman e-mail
AREA	Varchar	255		Salesman area

Name	Type	Length	Scale	Description
NO	Varchar	5		Invoice number
CUST	SmallInt	2	0	Customer code
SAL	SmallInt	2	0	Salesman code
DATE	TimeStamp	8		Order Date
TOTAL	Float	8	0	Amount

7. Define the join condition(s) if needed, and switch to the Query Tab. The created query appears as follow:

The screenshot shows the 'Query Tab' of the Click&DECiDE interface. The query structure is displayed in a tree view on the left, showing the 'salesman' and 'sales' tables. The main pane shows the query details, including the 'Name', 'Type', 'Length', and 'Scale' of the tables and their parameters.

Name	Type	Length	Scale	Description
salesman.parameters				
< caching >	Varchar	32	0	Caching mode: ON or empty (d
< scope >	Varchar	32	0	Caching scope: Empty: by use
< index >	Varchar	4000	0	Indexes: list of IX(IndexName, .
salesman				
SAL	Float	53	0	Salesman code
SALNAME	Varchar	255	0	Salesman name
MAIL	Varchar	255	0	Salesman e-mail
AREA	Varchar	255	0	Salesman area
sales.parameters				
< caching >	Varchar	32	0	Caching mode: ON or empty (d
< scope >	Varchar	32	0	Caching scope: Empty: by use
< index >	Varchar	4000	0	Indexes: list of IX(IndexName, .
sales				
NO	Varchar	5	0	Invoice number
CUST	SmallInt	2	0	Customer code
SAL	SmallInt	2	0	Salesman code
DATE	TimeStamp	8	3	Order Date
TOTAL	Float	8	0	Amount

8. The « salesman » table is coming from an Excel data source.
 9. The « sales » table is coming from a SQL Server data source.
 10. Each table appears with a « Parameter » level and a « Table » level.



11. Note that in the above example the queries « sales » and « salesman » did not contain any parameter already defined. This means that all data will be saved in the corresponding hidden tables according to the cache values.
12. Run the above query. Automatically one or several temporary tables containing the result are created in the Server SQL Server database.
13. We will explain later on that manually the incidence of the already existing parameters in the queries sources feeding a multi-data source query.

Parameter Level

This level contains some options to improve the performances:

<Caching> Mode:

The idea is to update the hidden tables through a Scheduled Task according to the specified cache.

- **ON** or empty (default): use or update cache, default value is 5 minutes. After that time the table is purged and deleted unless a scheduled task has recreated it.
- **OFF**: do not use cache.
- **<Positive integer>**: update cache if needed and keep for <integer> minutes. After that delay the table is purged and deleted unless a scheduled task has recreated it.
- **<Negative integer>**: always update cache and keep for <integer> minutes. You need to specify the Scope (Global recommended in that case).
- **permanent**: always update cache, keep 10 years (similar to the previous one but valid 10 years).

In the above example, we will define the cache as Permanent for both tables Salesmen and Sales:

<Scope>

The idea is to define who can use the result table. Global is recommended but empty means “by User” by default.

- Empty: by user by default.
- User: by user.
- Global: for all users.
- Memory: in memory for current session only.

In the above example, we will define the Scope as Global for both tables Salesmen and Sales:



<Index>

The idea is to be able to add one or several indexes on the table containing data in the cache, giving best performances to future queries.

- **list of IX**(IndexName, <fields>) = Index
 - IndexName = index name
 - <Field> = column name between 2 double-quotes if the name contains a space or special characters.
 - If several indexes, separate them with a semi-colon
- **UKC**(IndexName, <fields>) = Unique Index
 - IndexName = Unique Index name
 - <Field> = column name between 2 double-quotes if the name contains a space or special characters.

Example: IX(Index1, CUST); IX(Index1, "NO"); UKC(UniqueIndex1, SAL);

➤ Case of query source already containing one or several parameters

Supposing that one of the queries from the previous example was already containing a parameter on the DATE field with a list of period, such as keywords.



If we redo the multi data source query by joining the Excel (« salesman » table) with the SQL Server (« sales » table) and if this « Sales » query contains a parameter on the DATE field, we will get this result:

Name	Type	Length	Cache	Scope	Index	Description
salesman.parameters						
< caching>	Varchar	32	0	Caching mode: ON or empty (
< scope>	Varchar	32	0	Caching scope: Empty: by us		
< index>	Varchar	4000	0	Indexes: list of IX(IndexName,		
salesman						
SAL	Float	53	0	Salesman code		
SALNAME	Varchar	255	0	Salesman name		
MAIL	Varchar	255	0	Salesman e-mail		
AREA	Varchar	255	0	Salesman area		
sales.parameters						
< caching>	Varchar	32	0	Caching mode: ON or empty (
< scope>	Varchar	32	0	Caching scope: Empty: by us		
< index>	Varchar	4000	0	Indexes: list of IX(IndexName,		
P_DATE	Varchar	0	0			
sales						
NO	Varchar	5	0	Invoice number		
CUST	SmallInt	2	0	Customer code		
SAL	SmallInt	2	0	Salesman code		
DATE	TimeStamp	8	3	Order Date		
TOTAL	Float	8	0	Amount		

In the « parameters » branch for the « Sales » table we can see a new row with the P_DATE parameter. Although this parameter was a Date or Date Time Type, it appears here as a String because the value you can enter will be translated by the current multi data source query. For example, if you enter as criteria 2017, it will be translated with an interval including all dates for the year 2017. If you enter « Last Month » or « 1 months ago », it will be translated with an interval including all days of the last month etc.

If we add criteria on the left side of this parameter and if we run the query, only the records matching the criteria condition will be saved in the hidden temporary tables, according to the cache and scope values.

In case of Cache not defined, the default value will be 5 minutes and in case of Scope not defined, the default value will be "by user".

For example, we enter « Last Month » or « 1 months ago » as criteria for the P_DATE parameter:

Criteria

Columns: sales.parameters.P_DATE

Operators: =

Value: Last Month

Where: sales.parameters.P_DATE = 'Last Month'

Buttons: OK, Cancel, New, Remove, Parameters..., Not, Combine, Uncombine



Now we run the query and we get the records for the concerned month:

salesman

<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	SAL	Float	53	0	Salesman code
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	SALNAME	Varchar	255	0	Salesman name
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	MAIL	Varchar	255	0	Salesman e-mail
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	AREA	Varchar	255	0	Salesman area

sales.parameters

<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	< caching >	Varchar	32	0	Caching mode: ON or empty (default): use or update c
<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	< scope >	Varchar	32	0	Caching scope: Empty: by user or for all users (chose
<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	< index >	Varchar	4000	0	Indexes: list of IX(IndexName, < fields >) = Index or UKC
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	P_DATE	Varchar	0	0	

sales

<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	NO	Varchar	5	0	Invoice number
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	CUST	Smallint	2	0	Customer code
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	SAL	Smallint	2	0	Salesman code
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	DATE	TimeStamp	8	3	Order Date
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	TOTAL	Float	8	0	Amount

	SAL	SALNAME	MAIL	AREA	NO	CUST	sales.SAL	DATE	TOTAL
1	1	Bill Raley	bill.raley@yopmail.com	NORTH-WEST	16095	1003	1	12/22/2016	\$82,450.00
2	1	Bill Raley	bill.raley@yopmail.com	NORTH-WEST	16096	1003	1	12/23/2016	\$93,305.00
3	2	Sandra Davis	sandra.davis@yopmail.com	NORTH-WEST	16105	1010	2	12/31/2016	\$230,218.15
4	3	Jean Martin	jean.martin@yopmail.com	SOUTH	16091	1021	3	12/21/2016	\$89,207.00
5	3	Jean Martin	jean.martin@yopmail.com	SOUTH	16094	1021	3	12/22/2016	\$102,289.00
6	4	Diane Meyer	diane.meyer@yopmail.com	ATLANTIC	16101	1006	4	12/24/2016	\$197,881.50
7	7	Georges Dunel	georges.dunel@yopmail.com	WEST	16106	1005	7	12/31/2016	\$246,334.29

But what is important is to know that only those records will be saved into the hidden and temporary tables corresponding to this SQL query. That means that if in the cache delay duration, here 5 minutes, the query is run again with the same criteria, the response is immediate because data are coming from temporary tables and not from the original real data source.

If now we modify these criteria by entering for example « Since 3 months » the result will be saved in a new temporary distinct table for this query with these criteria, during the cache duration.

If we use other criteria or parameters on this multi data source query, then theses filters will apply to the data from the temporary tables as far as the cache value is valid. For example we can add a parameter on the AREA field in the above query:

Criteria

Columns

salesman.AREA

Operators

=

Value

Column

Expression

Query

P_AREA

Where

sales.parameters.P_DATE = 'Last Month'

and

salesman.AREA = P_AREA

Not

Combine

Uncombine

OK

Cancel

New

Remove

Parameters...

If we run, we get a question about the AREA:



Parameters

☒ Choose value for AREA

ATLANTIC
CENTRAL
NORTH-WEST
SOUTH
WEST

OK
Cancel

✓ ✗

And if we select any Area, the data are still coming from the temporary tables, only for that area, as far as the cache value is still valid for the DATE parameter:

AREA	Varchar	255	0 Salesman area
sales.parameters			
< caching >	Varchar	32	0 Caching mode: ON or empty (default): use or update c
< scope >	Varchar	32	0 Caching scope: Empty: by user or for all users (chose
< index >	Varchar	4000	0 Indexes: list of IX(IndexName, <fields>) = Index or UKC
P_DATE	Varchar	0	0
sales			
NO	Varchar	5	0 Invoice number
CUST	SmallInt	2	0 Customer code
SAL	SmallInt	2	0 Salesman code
DATE	TimeStamp	8	3 Order Date
TOTAL	Float	8	0 Amount

	SAL	SALNAME	MAIL	AREA	NO	CUST	sales.SAL	DATE	TOTAL
1	1	Bill Raley	bill.raley@yopmail.com	NORTH-WEST	16095	1003	1	12/22/2016	\$82,450.00
2	1	Bill Raley	bill.raley@yopmail.com	NORTH-WEST	16096	1003	1	12/23/2016	\$93,305.00
3	11	Joe Kramer	joe.kramer@yopmail.com	NORTH-WEST	16100	1010	11	12/24/2016	\$186,527.50
4	11	Joe Kramer	joe.kramer@yopmail.com	NORTH-WEST	16102	1010	11	12/26/2016	\$200,114.50
5	2	Sandra Davis	sandra.davis@yopmail.com	NORTH-WEST	16105	1010	2	12/31/2016	\$230,218.15

In summary, in the above query, we need to remember that:

The criteria defined on the **sales.parameters** branch level for the **P_DATE** parameter will modify the number of records saved in the temporary tables during all the valid cache period for this query.

The criteria defined on the **AREA** field level is used as a **filter on the result** extracted from these temporary tables.



2- Create and use synonyms with « Click and DECiDE Queries » data source:

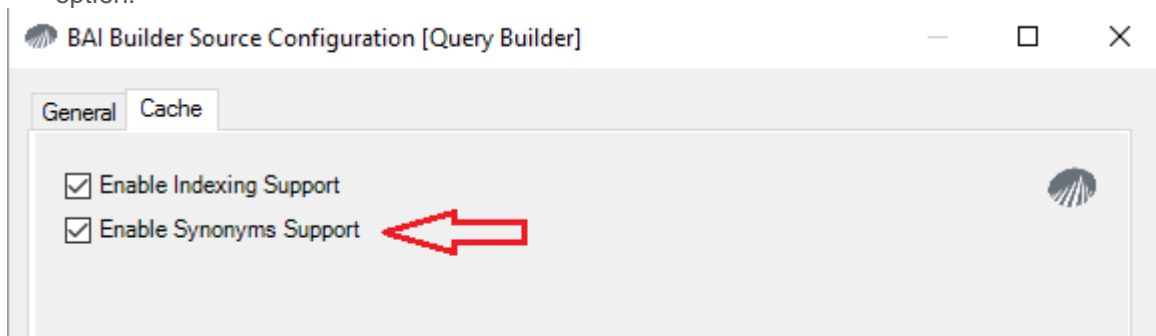
- In this chapter, we will describe how to use this new « Click and DECiDE Queries » data source to create some “synonyms”, from single or multi data sources queries.
- The synonyms are some « views » pointing to temporary tables set in cache, according several options such as cache, scope and indexes.
- The end-users will use a standard SQL Server data source to extract data from these synonyms.
- Note that in the synonyms, some of them can be created from an iSeries Data Source, other from an Excel or SQL Server data source etc., and that the end-user working with these synonyms can make queries multi data source without knowing it.

Use :

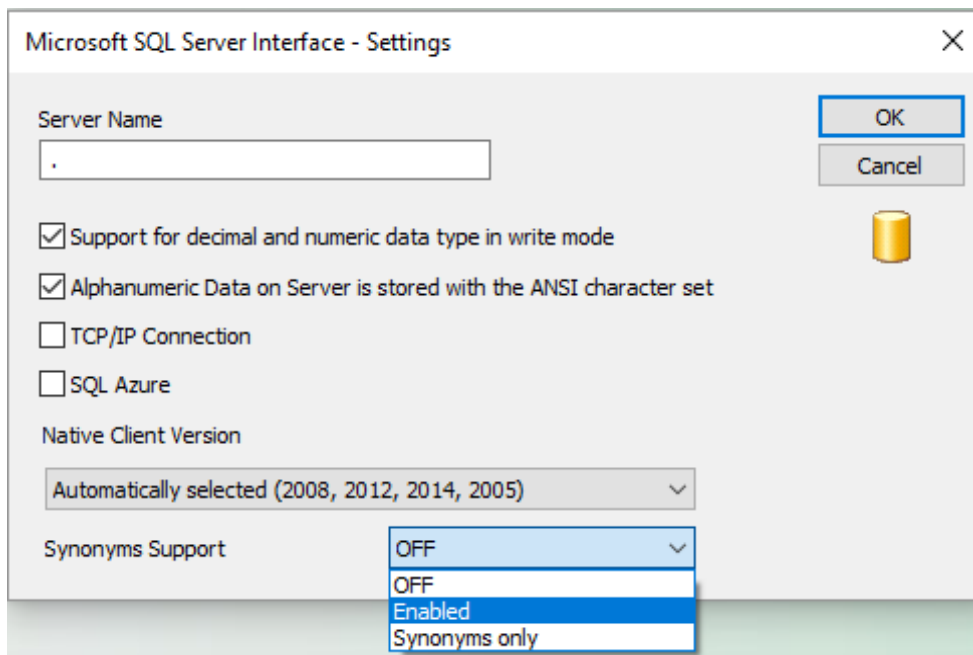
1- Create a Click and DECiDE Queries data source:

In this step already describe in point 1, you just only need to activate the synonyms in the Click&DECiDE Queries data source configuration:

- In the « Cache » Tab of this Click&DECiDE data source properties in Administration Manager, activate this option:



- Warning: you also need to check if the SQL Server data sources you will use on that Server where Click&DECiDE Enterprise Edition is installed have another option enabled to support the synonyms management (Local SQL Server and/or Secured SQL Server).
- Open the data source properties in Click&DECiDE Administration Manager :





- ✓ « Enabled » option will allow the end-users to access the synonyms, but also access to tables and views.
- ✓ The « Synonyms only » option will limit the visibility to the synonyms only. If no synonyms exist, Builder will display nothing when browsing the tables list.

Save this configuration and click the command File > Update Security.

Use from Click&DECiDE Builder

➤ Query source containing or not containing parameters.

1. Start Builder (or Web Builder from the Web Portal)
2. Open or create a project (for example the file My Queries Multi Sources.wfv).
3. If the « Query Builder » data source has been authorized for the User Group you belong to, you should see the list of the projects authorized for that User Group, or for these Groups if you belong to several User Groups.
4. Create a new query from this « Query Builder » data source:
Select the required project to display the existing queries, as tables sources, for example the project « iSeries queries » containing iSeries queries (AS/400):

In this example we will use an iSeries query containing already 2 parameters:

A P_DATE parameter on the DATE field.

A P_AREA parameter on the AREA field.

Parameter Properties	
Name	P_DATE
Type	Timestamp
Status	Enable
Update Method	Input List
Title	Choose value for DATE
Mandatory	No
Editable	No
Default Value	IGNORE
Value List	in 2 days Day after tomorrow;in 1 days Tomorrow;0 days ago Today;1 days ag...
Selection	Single
Hide List's First Column	Yes

This query is saved under the name « iSeries Sales » in the iSeries Queries.wfv project file.

Create this new query selecting that project file iSeries Queries.wfv and the query named « iSeries Sales »:



Content of [Query Builder]: "iSeries Queries"

Name	Description
iSeries Sales	Sales Table on AS/400
Julian and week	
query	
Test Julian and Week	
week istone	

iSeries Sales

Name	Type	Length	Scale	Description
NO	Varchar	5		Invoice Number
CUST	SmallInt	2	0	Customer Code
SAL	SmallInt	2	0	Salesman Code
SALNAME	Varchar	15		Salesman Name
AREA	Varchar	10		Sales Area
DATE	TimeStamp	26		Order Date
TOTAL	Float	31	8	Amount

Go to the « Query » Tab and select the required columns:

My Queries Mutli Sources.wfv* Query*

Name	Type	L	
iSeries Sales.parameters			
< caching >	Varchar	32	0 Caching mode: ON or empty
< scope >	Varchar	32	0 Caching scope: Empty: by u
< index >	Varchar	4000	0 Indexes: list of IX(IndexNam
< synonym >	Varchar	256	0 Synonym for the data table :
P_DATE	Varchar	0	0
P_AREA	Varchar	0	0
iSeries Sales			
<input checked="" type="checkbox"/> NO	Varchar	5	0 Invoice Number
<input checked="" type="checkbox"/> CUST	SmallInt	2	0 Customer Code
<input checked="" type="checkbox"/> SAL	SmallInt	2	0 Salesman Code
<input checked="" type="checkbox"/> SALNAME	Varchar	15	0 Salesman Name
<input checked="" type="checkbox"/> AREA	Varchar	10	0 Sales Area
<input checked="" type="checkbox"/> DATE	TimeStamp	26	0 Order Date
<input checked="" type="checkbox"/> TOTAL	Float	31	8 Amount

Invoice Number Customer Code Salesman Name Sales Area Order Date Amount



The options such as Cache, Scope and optional index have already been described in the first part of this manual. **Nevertheless, note that, when synonyms are used, the cache must be defined as « permanent » and the Scope as « Global ».** We can add (optional) an index on the DATE field and a unique index on the « NO » field (Invoice Number):

```
IX(Index1, ""DATE""); UKC(UniqueIndex1, ""NO"");
```

<Synonyms>

The idea is to create a view pointing on a temporary result table that will be available for the end-users who will not be managing themselves these caches, scopes and indexes. The end-users will see and will use the synonyms as if it was some SQL Server data source tables.

We will thus define here a customized name for that synonym, that means a “view” name in SQL Server.

To do so, click the criteria square on the left of the « synonyms » parameter in the **iSeries Sales.parameters** and enter as name for example **DataSetReport.demo.Sales 2015 North-West**, or **<Database Name.Schema name.synonym name>**.



The screenshot shows the 'Criteria' dialog box. The 'Columns' dropdown is set to 'iSeries Sales.parameters.<synonym>'. The 'Operators' dropdown is set to '='. The 'Value' field contains 'DataSetReport.demo.Sales 2015 North-West'. The 'Where' section contains a list of criteria separated by 'and':

- iSeries Sales.parameters.< caching> = 'permanent'
- iSeries Sales.parameters.< scope> = 'Global'
- iSeries Sales.parameters.< index> = 'IX(Index1, ""DATE""; UKC(UniqueIndex1, ""NO"";'
- iSeries Sales.parameters.< synonym> = 'DataSetReport.demo.Sales 2015 North-West'

Buttons at the bottom include OK, Cancel, New, Remove, and Parameters...

Validate with OK.

Also note that, in the **iSeries.Sales.parameters**, we can see that the parameters P_DATE and P_AREA from the query source are displayed. Of course if the query source contains no parameters, ignore this paragraph. If we add criteria on these parameters, it will limit the content of the temporary tables that will be updated when running this query, and according to the options given for the Cache, the Scope and the optional indexes, and depending of the criteria values.

To limit the records to the year 2015 for example and to the 'North-West' area, we can add the following criteria for the corresponding parameters:

- On the left side of P_DATE parameter enter as criteria 2015.
- On the left side of P_AREA parameter enter as criteria NORTH-WEST.

In fact, all criteria should appear as follow:

The screenshot shows the 'Criteria' dialog box with the same criteria as the previous image, plus two additional criteria:

- iSeries Sales.parameters.P_DATE = '2015'
- iSeries Sales.parameters.P_AREA = 'NORTH-WEST'

The 'Where' section now contains six criteria separated by 'and'. The 'Value' field still contains 'NORTH-WEST'. Buttons at the bottom include OK, Cancel, New, Remove, and Parameters...



Validate with OK and run the query.

The screenshot shows the 'Query*' window in Click&DECiDE. It displays a multi-source query configuration with two main sections: 'iSeries Sales.parameters' and 'iSeries Sales'. Below the configuration, a table shows the results of the query.

	Name	Type			
iSeries Sales.parameters					
<input type="checkbox"/>	< caching >	Varchar	32	0	Cachi
<input type="checkbox"/>	< scope >	Varchar	32	0	Cachi
<input type="checkbox"/>	< index >	Varchar	4000	0	Index
<input type="checkbox"/>	< synonym >	Varchar	256	0	Synou
<input type="checkbox"/>	P_DATE	Varchar	0	0	
<input type="checkbox"/>	P_AREA	Varchar	0	0	
iSeries Sales					
<input checked="" type="checkbox"/>	NO	Varchar	5	0	Invoic
<input checked="" type="checkbox"/>	CUST	SmallInt	2	0	Custo
<input checked="" type="checkbox"/>	SAL	SmallInt	2	0	Sales
<input checked="" type="checkbox"/>	SALNAME	Varchar	15	0	Sales
<input checked="" type="checkbox"/>	AREA	Varchar	10	0	Sales
<input checked="" type="checkbox"/>	DATE	TimeStamp	26	0	Order
<input checked="" type="checkbox"/>	TOTAL	Float	31	8	Amou

	Invoice Number	Customer Code	Salesman Name	Sales Area	Order Date	Amount
1	15001	1003	Bill Raley	NORTH-WEST	1/4/2015	\$94,029.60
2	15012	1011	Robert Salta	NORTH-WEST	2/2/2015	\$107,565.00
3	15016	1011	Robert Salta	NORTH-WEST	2/4/2015	\$122,289.50
4	15019	1010	Joe Kramer	NORTH-WEST	3/6/2015	\$70,283.20
5	15020	1010	Joe Kramer	NORTH-WEST	3/7/2015	\$80,889.60
6	15021	1011	Robert Salta	NORTH-WEST	3/9/2015	\$44,940.00
7	15032	1003	Bill Raley	NORTH-WEST	4/6/2015	\$79,845.00
8	15033	1003	Bill Raley	NORTH-WEST	4/7/2015	\$90,069.50

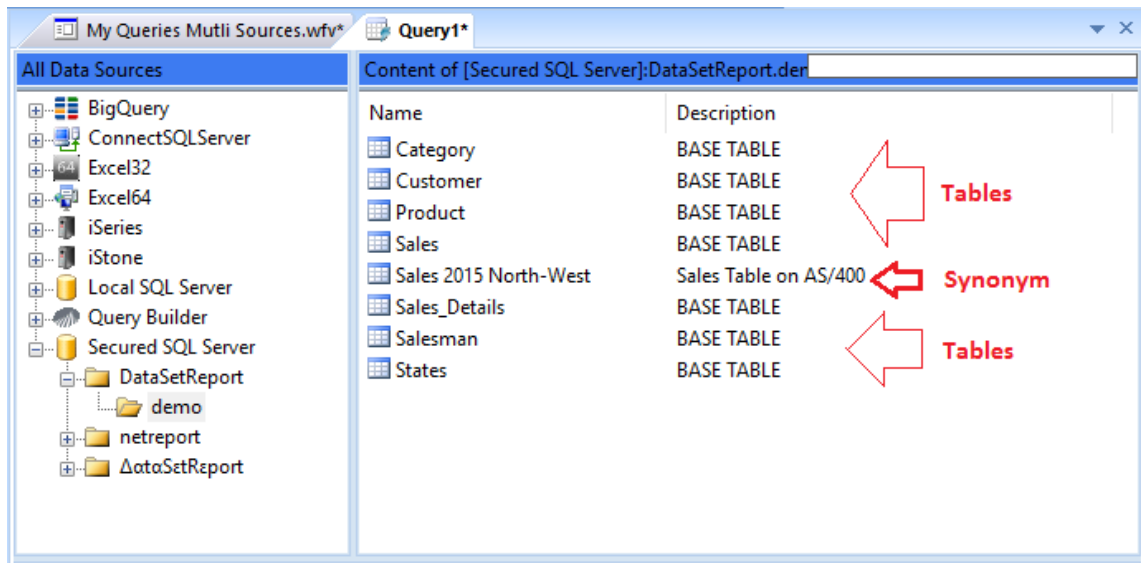
On the background, automatically a synonym named « Sales 2015 North-West » has been created in SQL Server and that synonym is pointing to a temporary table only containing the records for the year 2015 and for the North-West area.

How to make a query on that synonyms?

From Click&DECiDE Builder (or from Click&DECiDE Web Builder through the Web Portal), create a new query by selecting the « Local SQL Server » data source or « Secured SQL Server » data source if through the Web Portal, as in that case the data source must be defined as “Secured by Click and DECiDE”.

Be sure that both data sources have the « Synonyms support » option enabled in Click&DECiDE Administration Manager.(See the paragraph « how to activate the synonyms).

For example, create a new query on the Secured SQL Server secured data source on the database and schema declared in the full synonym name: (DataSetReport.demo)



Among the tables list from this DataSetReport database we can see the new synonym that has been created when running the previous query. Select this synonym and select all fields then run the query: the data are corresponding to the year 2015 and the North-West area declared as criteria on the P_DATE and P_AREA parameters.

Name	Type	
NO	Varchar	5 0 Invoice Number
CUST	Integer	2 0 Customer Code
SAL	Integer	2 0 Salesman Code
SALNAME	Varchar	15 0 Salesman Name
AREA	Varchar	10 0 Sales Area
DATE	TimeStamp	8 3 Order Date
TOTAL	Float	8 0 Amount

	Invoice Number	Customer Code	Salesman Code	Salesman Name	Sales Area	Order Date	Amount
1	15001	1003	1	Bill Raley	NORTH-WEST	1/4/2015	\$94,029.60
2	15012	1011	8	Robert Salta	NORTH-WEST	2/2/2015	\$107,565.00
3	15016	1011	8	Robert Salta	NORTH-WEST	2/4/2015	\$122,289.50
4	15019	1010	11	Joe Kramer	NORTH-WEST	3/6/2015	\$70,283.20
5	15020	1010	11	Joe Kramer	NORTH-WEST	3/7/2015	\$80,889.60
6	15021	1011	8	Robert Salta	NORTH-WEST	3/9/2015	\$44,940.00

On that query result, you can now apply other criteria or parameters as in a standard Builder query.

Note that, to simplify, this manual has been done on a single data source query to explain the synonyms feature, but it is of course possible to do the same using a multiple data source query.

If for example, you have created several synonyms on various data sources, it is thus possible for the end-users to create a new standard SQL Server query by joining several synonyms together.

Example: the synonym created in this manual is named « Sales 2015 North-West » and is corresponding to data coming from an iSeries.



We can imagine another synonym named « North-West Salesmen » corresponding to data managed in another data source Oracle, then a query joining these 2 synonyms could be similar to a multiple data source query but without the need to manage technical options such as caches, scopes or indexes reserved to advanced users.

Conclusion:

The Click and DECIDE Queries data source is thus mainly reserved to Administrators, or advanced users, while the Synonyms access is mainly reserved to end-users through a data warehouse.

***** End of documentation *****