



QlikView 11.2 DIRECT DISCOVERY

QlikView Technical Addendum

Published: November, 2012

Version: 3.0

Last Updated: December, 2012

www.qlikview.com

A decorative graphic in the top left corner of the page, consisting of several overlapping, flowing green lines that create a sense of movement and depth. The lines are thin and have a slight transparency, allowing them to overlap and create darker shades of green.

Overview

This document provides a technical overview of the QlikView 11.2 Direct Discovery feature.

QlikView Direct Discovery capability expands the potential use cases for Business Discovery, enabling business users to conduct associative analysis on big data sources. It provides QlikView's complete associative experience on top of data coming directly from external big data sources, and enables users to combine that big data with data stored in memory. With QlikView Direct Discovery, business users can leverage any data useful for analysis without scalability limitations.

The following part of the paper provides a technical overview of implementing and using QlikView Direct Discovery. It also provides information on the best practices and limitations of this feature with this release.

What is QlikView Direct Discovery

QlikView Direct Discovery capability combines the associative capabilities of the QlikView in memory dataset with a query model where the source data is not directly loaded into the QlikView data model, instead the aggregated query result is passed back to QlikView user interface. The direct discovery data set is still part of the associative experience where the user can navigate both on the in-memory data and the direct discovery data associatively.

The business users can associatively make selections on either of the data sets, and see what is associated and not associated with the same QlikView association colours; green, gray, and white. They can create charts that would help them analyze data from both data sets together.

Technical Details

With QlikView's unique associative experience in combination with Direct Discovery users are able to navigate and interact with data by a combination of methods.

I. HOW DOES DIRECT DISCOVERY WORK

QlikView determines which data resides in-memory and which data is direct discovery data by using the special script syntax, "DIRECT SELECT". This syntax allows certain data elements not to be loaded into the QlikView data model during the script reload process, but still available for query purposes from the QlikView User Interface and to be combined for analysis with the QlikView in memory dataset.

Once the direct discovery structure is established, the direct discovery fields can be used with certain QlikView objects. When a direct discovery field is used on the QlikView object, QlikView will automatically create the appropriate SQL query to run on the external data source. The result of the query will be displayed on the QlikView object. When selections are made on the QlikView application, the associated data values of the direct discovery fields will be used in the WHERE conditions of the queries. With each selection, the direct discovery charts will be calculated, where the calculations and aggregations will be done on the source table by executing the SQL query created by QlikView. It is possible to use calculation condition feature of the QlikView charts to set a condition indicating when the chart should be calculated. Until that condition is met, QlikView will not run queries and the chart will not be calculated. Please note that QlikView will execute SQL queries on the data source for some of the list boxes that use direct discovery fields. This is required to achieve the associative navigation capability.

II. DATA LOADING

Within the script editor a new syntax is introduced to connect to data in direct discovery form. Traditionally the following syntax is required to load data from a database table:

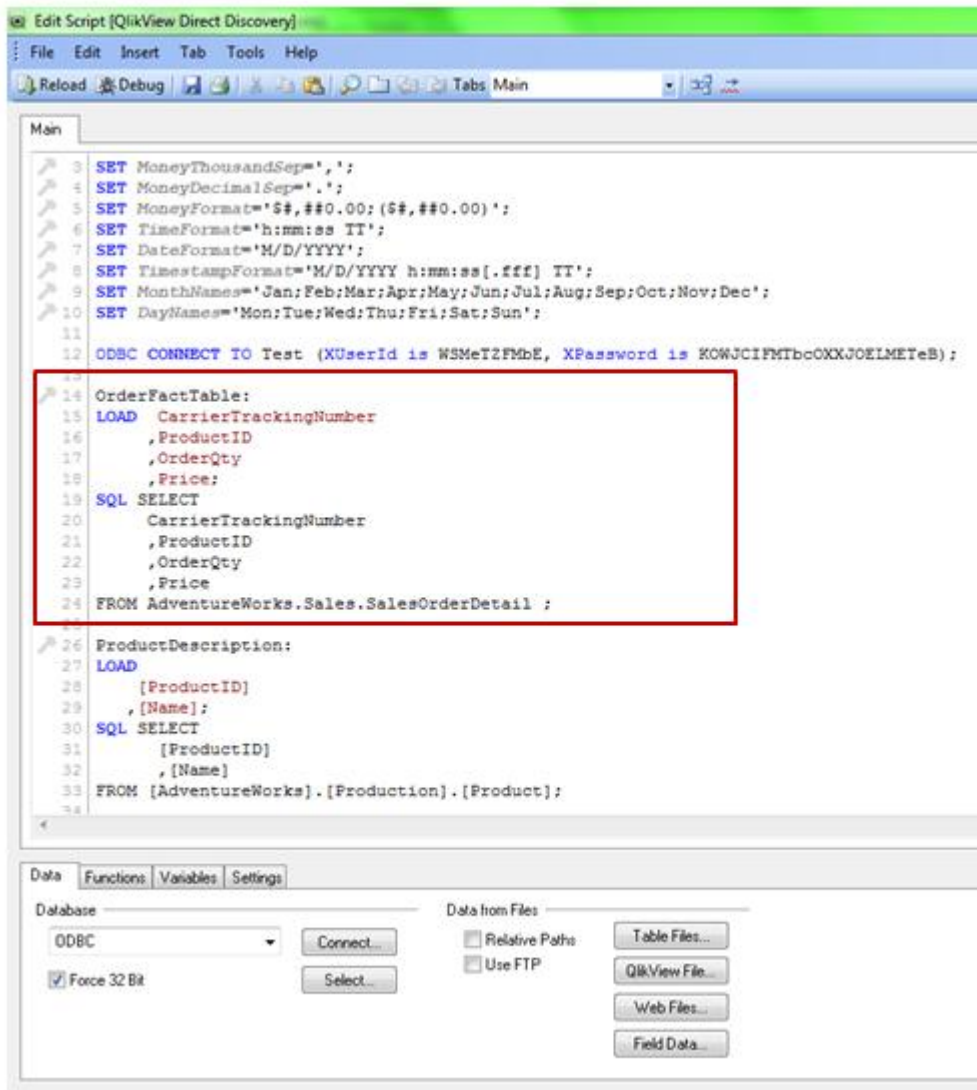


Figure 1. Traditional QlikView Load Script Syntax

To invoke the direct discovery method, the keyword “SQL SELECT” is replaced with “DIRECT SELECT”:

```

2 SET DecimalSep='.';
3 SET MoneyThousandSep=',';
4 SET MoneyDecimalSep='.';
5 SET MoneyFormat='$#,##0.00;($#,##0.00)';
6 SET TimeFormat='h:mm:ss TT';
7 SET DateFormat='M/D/YYYY';
8 SET TimestampFormat='M/D/YYYY h:mm:ss[.fff] TT';
9 SET MonthNames='Jan;Feb;Mar;Apr;May;Jun;Jul;Aug;Sep;Oct;Nov;Dec';
10 SET DayNames='Mon;Tue;Wed;Thu;Fri;Sat;Sun';
11
12 ODBC CONNECT TO Test (XUserId is WSMetZFMbE, XPassword is KOWJCIFMTbcOXXJOELMETeB);
13
14 OrderFactTable:
15 DIRECT SELECT
16     CarrierTrackingNumber
17     ,ProductID
18 FROM AdventureWorks.Sales.SalesOrderDetail ;
19
20 ProductDescription:
21 LOAD
22     [ProductID]
23     , [Name];
24 SQL SELECT
25     [ProductID]
26     , [Name]
27 FROM [AdventureWorks].[Production].[Product];
28

```

Figure 2. QlikView Load Script Syntax for Direct Discovery

In the example above, the source data table “SalesOrderDetail” has 4 fields; CarrierTrackingNumber, ProductID, OrderQty and Price. With the use of “DIRECT SELECT” keyword, only column CarrierTrackingNumber and ProductID are loaded in memory as symbol tables (they are referred as explicit direct discovery fields). These two columns are loaded in memory to create the associations between the in-memory data and the direct discovery data. These fields can also be used for selection purposes on the QlikView user interface. Other columns, OrderQty and Price, exist in the source data table within the database and they are not part of the in memory data model. OrderQty and Price fields are referred as “IMPLICIT” fields. An implicit field is a field that QlikView is aware of on a “meta level”. The actual data of an implicit field resides only in the database but the field may be used in QlikView expressions. More information on the implicit fields is provided in the next section. Please note that preceding load cannot be used with Direct Discovery as it only relates to the data that is loaded in memory.

Looking at the table view and data model, the implicit direct discovery fields are not within the model (DirectTable is the table created on the data model with the explicit direct discovery fields. This table allows QlikView to query the source database table as users make selections on the user interface):

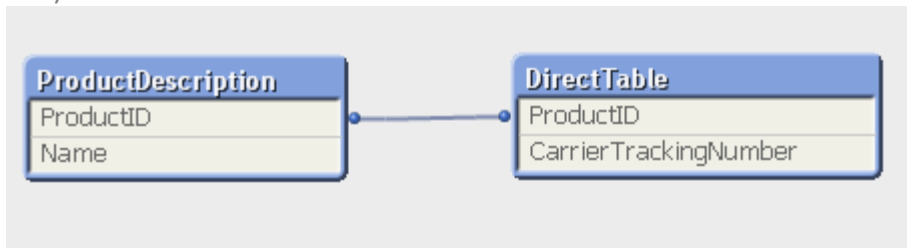


Figure 3. Associative Data Model for Direct Discovery and In-memory data sets

Once the direct discovery structure is established, the direct discovery data can be joined with the in-memory data with the common field names (Figure 3). In this example, “ProductDescription” table is loaded in memory and joined to the direct discovery data with the ProductID field. This allows the user to associatively navigate both on the direct discovery and in memory data sets. As figure 4 demonstrates, the business users can associatively make selections on either of the data sets (direct discovery or in-memory), and see what is associated and not associated with the same QlikView association colors; green, gray, and white.

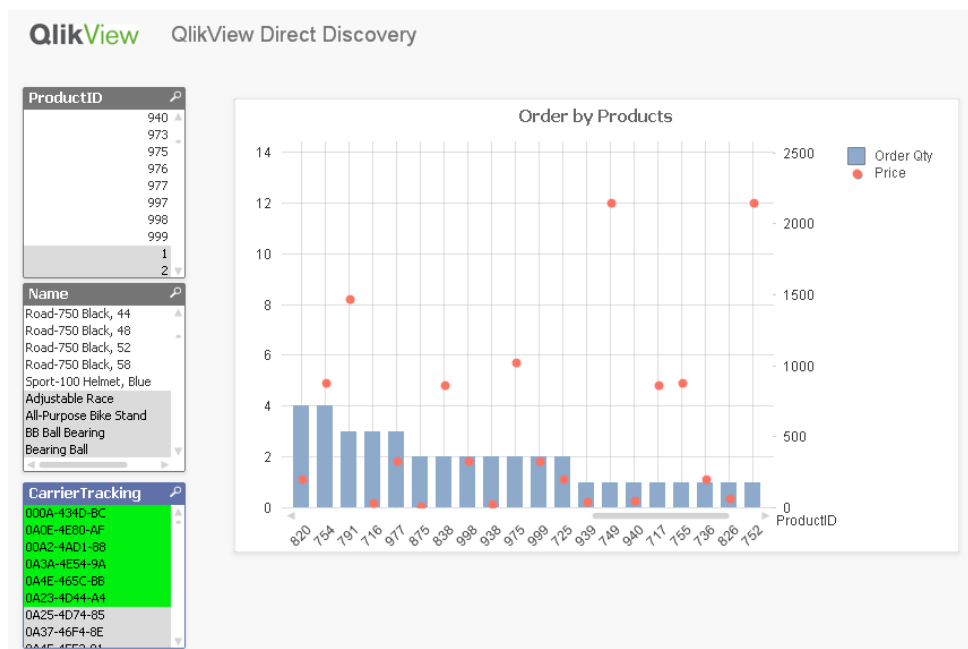


Figure 4. Associative Business Discovery on direct discovery and in-memory data sets

Please note that when using SQL functions with the direct discovery table on the load script, the SQL function should be used within single quotation marks.

```

15 DIRECT SELECT
16     CustomerID
17     ,SalesPersonID
18     ,SalesOrderID
19     ,OrderDate
20     , 'month([OrderDate])' as OrderMonth
21     , 'year([OrderDate])' as OrderYear
22 FROM AdventureWorks.Sales.SalesOrderHeader;

```

Figure 5. Use of SQL function with the direct discovery table on the load script

This initial version of direct discovery only supports one direct discovery table per QlikView application. Also, the source of the direct discovery can be one database table or one view.

Background and definition of implicit fields

An implicit field is a field that QlikView is aware of on a “meta level”. When QlikView recognizes the keyword “DIRECT SELECT” on the load script, it only automatically loads the metadata of the fields from the source table (even though the fields are not listed on the load script), and makes them “IMPLICIT” fields. The actual data of an implicit field resides only in the database but the field may be used in QlikView expressions. The idea is that an implicit field will be treated as any other field when the user works with expressions in QlikView. Please note that with this release of direct discovery, the implicit fields are required using an aggregation function when used on the user interface. When QlikView generates queries including implicit direct discovery fields, it uses a GROUP BY on the SQL statement by using the appropriate dimensionality to improve the query performance. If the requirement is to display a non-metric field (such as description) as an implicit direct discovery field, it would require using some type of aggregation function with the field (e.g.: Min(), Max(), or some type of text SQL function applicable with the direct discovery data source).

The reason to introduce the concept of implicit fields is for QlikView to decide if an aggregation should be done on the database instead of being processed by QlikView itself.

For example, if the expression Count(Address) is used in a QlikView chart and Address is an implicit field, QlikView will let the database do the count (obviously since QlikView doesn’t have any data for the Address field). To put it in another way; implicit fields will add support for aggregating in the database while using the same expression syntax QlikView already has. The result is that the usage of the direct discovery fields from the database will be transparent for the user.

For more advanced usages or database specific tasks, the user may utilize the new SQL(“) function (Figure 6) in the QlikView chart expressions. This new function would allow the user to run SQL functions, which are compliant with the direct discovery data source, directly from QlikView expression definition (e.g.: SQL(‘avg(UnitPrice)’)). Please note that the SQL statements can only be created against the database table that is defined as Direct Discovery table on the load script. When SQL(“) is used in the QlikView chart expression, the expression totals will not be displayed and the use of snowflake dimensions on the chart is not supported.

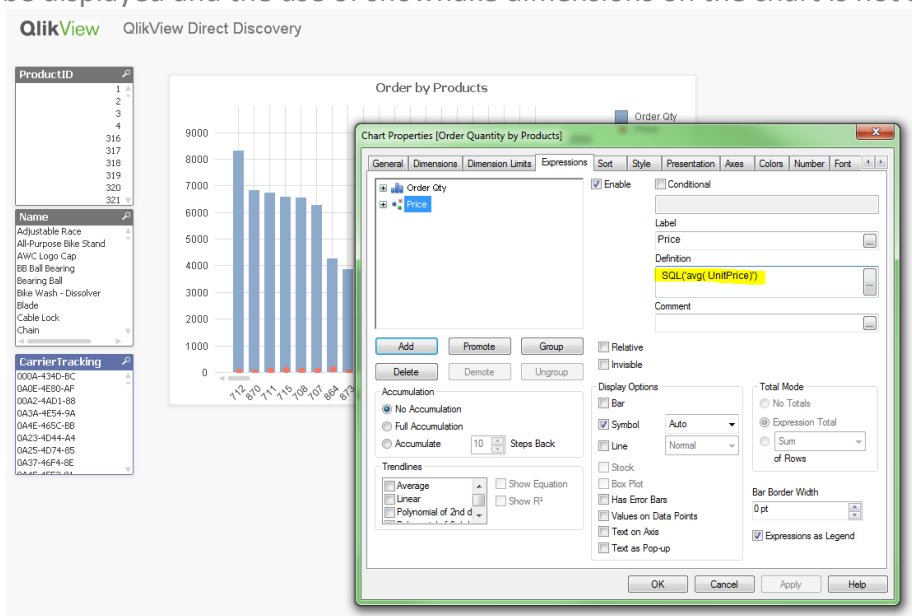


Figure 6. The use of SQL() function in the chart expression – SQL(‘avg(UnitPrice)’)

How to define implicit fields

There are two ways of defining implicit fields. The most straightforward way is to only load the fields which will be used for selections purposes or to join the direct discovery data to other in memory data. All other fields of the database table will be traded as IMPLICIT fields. On direct discovery table script load, QlikView will fetch all of the other fields from the database table and store the knowledge of them as implicit fields.

The second way is to use the keyword “IMPLICIT” on the load script. Once this keyword is used, QlikView will only load the fields listed as IMPLICIT in addition to the fields listed after the keyword “DIRECT”. Only these fields from the database will be available on the QlikView

application user interface for Business Discovery. This scenario can be used by the developer in the cases where there are fields on the database table that should be hidden from the user.

Please see examples below for these two scenarios.

Examples

Database fields: Field1, Field2, Field3, Field4, Field5

- **DIRECT SELECT Field1, Field3 FROM TableName**
Field2, Field4 and Field5 will be known as implicit fields to QlikView
- **DIRECT SELECT Field1, Field3 IMPLICIT Field2, Field4 FROM TableName**
Only Field2 and Field4 will be known as implicit fields to QlikView

It is possible to rename the IMPLICIT fields on the load script. This would require using "IMPLICIT" keyword.

DIRECT SELECT Field1, Field3 IMPLICIT Field2 AS FieldB FROM TableName

Field2 is renamed to FieldB and only FieldB will be known as implicit fields to QlikView

Explicit direct discovery fields

Explicit direct discovery fields are the fields that are listed after DIRECT SELECT on the load script. The unique data values of these fields are loaded in memory data model as symbol tables. The main use for explicit fields are;

- To create the associations between the in-memory data and the direct discovery data
- To define list boxes with direct discovery data (only explicit fields can be used with list boxes) when the requirement is to display direct discovery data for selection purposes on the QlikView user interface

Please note that when the explicit direct discovery fields are used in list boxes, the data values displayed in the list boxes will not get updated with direct queries. The only way to update the explicit direct discovery fields' data values on a list box is reloading the QlikView application.

III. CREATING QLIKVIEW OBJECTS WITH DIRECT DISCOVERY FIELDS

Once the direct discovery structure is setup, the direct discovery fields from the source table will be available on the user interface. Please note that only fields specified after the DIRECT SELECT statement (explicit fields) will be loaded into QV in the traditional manner. These fields can be used for selections or to enable associations between in memory data fields and direct discovery fields. All of the other fields from the data source will be available as implicit fields.

Due to the interactive and SQL specific nature of QlikView Direct Discovery, only certain QlikView objects can use implicit direct discovery fields. Table box, statistics box, pivot and mini charts are not supported with implicit direct discovery fields.

On the field list, additional information is provided to notify the user that the field is an implicit field (figure 7). This is useful information as some of the implicit fields from the big data sources may have billion values and using these fields on the user interface may slow down the user experience. Detailed attention should be paid with the use of implicit fields. Although the data values for the implicit fields are not loaded in memory, they still consume memory and CPU once they are used on the user interface of a QlikView application.

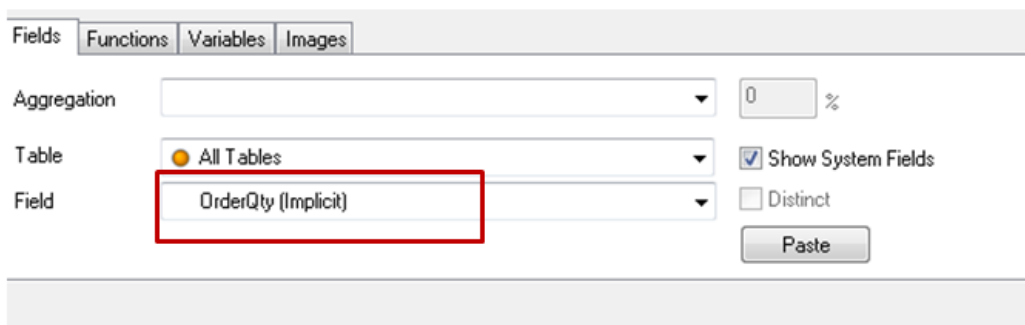


Figure 7. The direct discovery fields are marked with the keyword “Implicit” on the field list.

How to create list boxes with direct discovery fields

The traditional way of creating list box is used to create list boxes with the explicit direct discovery fields (fields that are listed after “DIRECT SELECT” keyword on the load script).

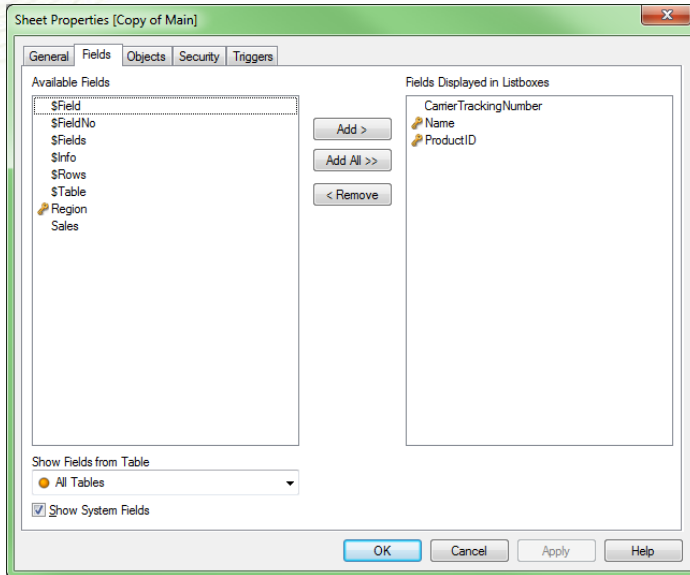


Figure 8. Traditional way of creating list boxes

In the previous example, CarrierTrackingNumber and ProductID fields can be used in list boxes as these are the explicit direct discovery fields.

Please note that to achieve the associative navigation capability; there will be cases where QlikView will execute SQL queries on the direct discovery data source when there is a selection made on the list box using a direct discovery field.

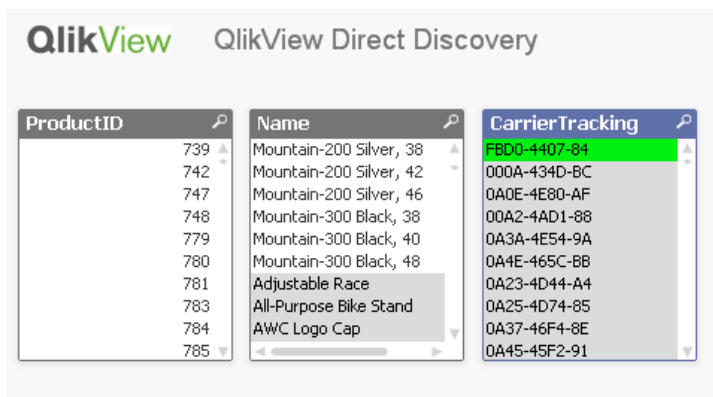


Figure 9. Associative data discovery with list boxes using direct discovery and in-memory data fields

It is possible to use the expression option of the list box with an implicit direct discovery field. The `aggr()` function should be used to show the aggregated value of the implicit field for any dimension field. For example, the list box expression “`aggr (sum (OrderQty) ,Name))`” will list the

A decorative graphic in the top left corner of the page, consisting of several overlapping, flowing green lines that create a sense of movement and depth.

aggregated OrderQty values for product names in a list box (OrderQty is an implicit direct discovery field).

How to create QlikView charts with direct discovery fields

The traditional way of creating charts is used to create a chart using direct discovery fields. It is possible to use the direct discovery fields as dimensions and/or expressions on the charts. However, please note that only implicit direct discovery fields can be used in expressions and explicit direct discovery fields can be used as dimensions. In the previous example, only CarrierTrackingNumber, ProductID fields can be used as chart dimensions and OrderQty, Price fields can be used as chart expressions.

For the QlikView charts that use only direct discovery fields, all of the aggregations are done on the database. If the QlikView chart has fields both from the in-memory and direct discovery tables, a second level aggregation is done on the chart level for the in memory fields once the database level aggregations are done.

As QlikView Direct Discovery generates SQL as its base code, not all QlikView expression functionality will be compatible with the feature. The expression functions that are supported with this initial release are; Sum, Avg, Count, Min, Max. Other functionality may be available pending time frames. Please note that it would be important to consider the type of aggregations that the source database supports when using direct discovery. For example, most SQL database supports DISTINCT in any aggregation, but Google BigQuery only supports COUNT(DISTINCT ...).

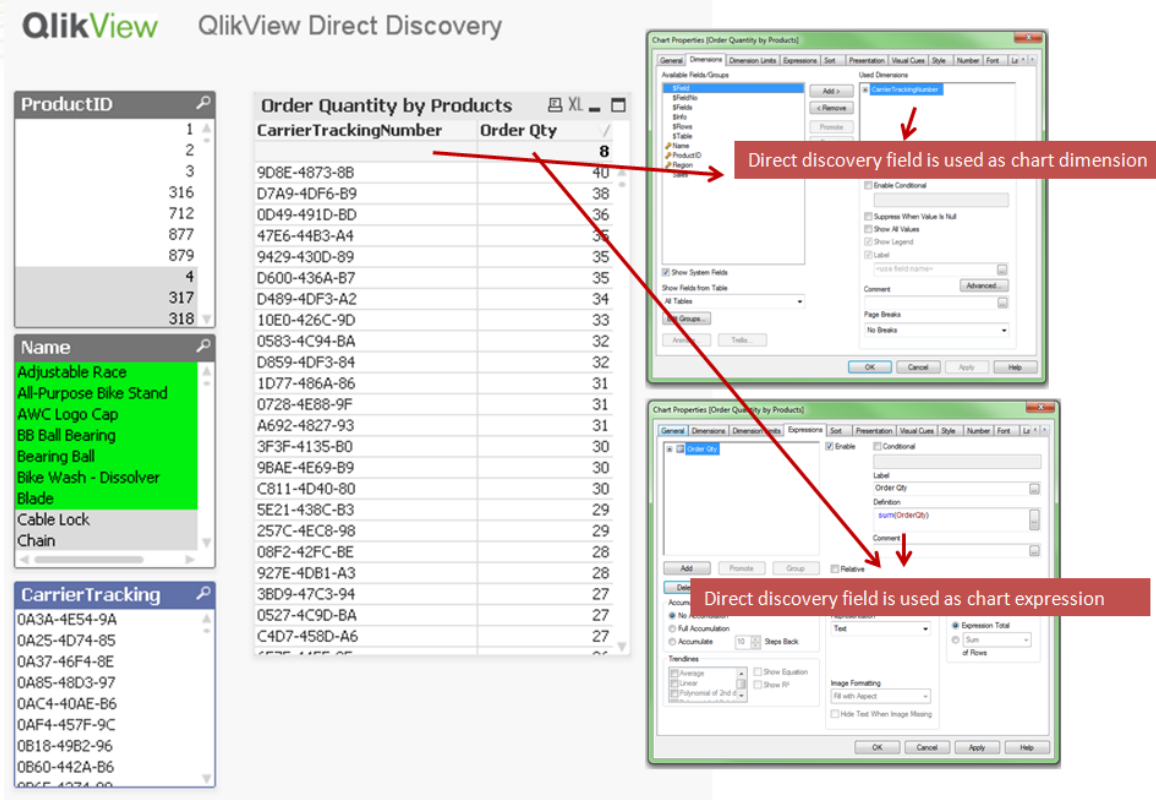


Figure 10. QlikView chart using direct discovery fields as dimension and expression

Most of the QlikView chart functionalities (interactive sorting, formatting, visual clues, dimension limits etc...) are still available to be used on the charts that use direct discovery fields. This provides the capability to leverage the same ease of use and rapid app development QlikView experience on the big data analysis.

Due to the SQL syntax specific nature of the direct discovery, pivot tables and mini charts are not supported with the use of direct discovery fields.

It is also possible to use the in memory fields and direct discovery fields on the same chart. In these cases, QlikView associative technology automatically handles the associations between the in memory fields and direct discovery fields. The business user still has the same easy to develop QlikView experience and does not need to worry about how the data sets are joined together.

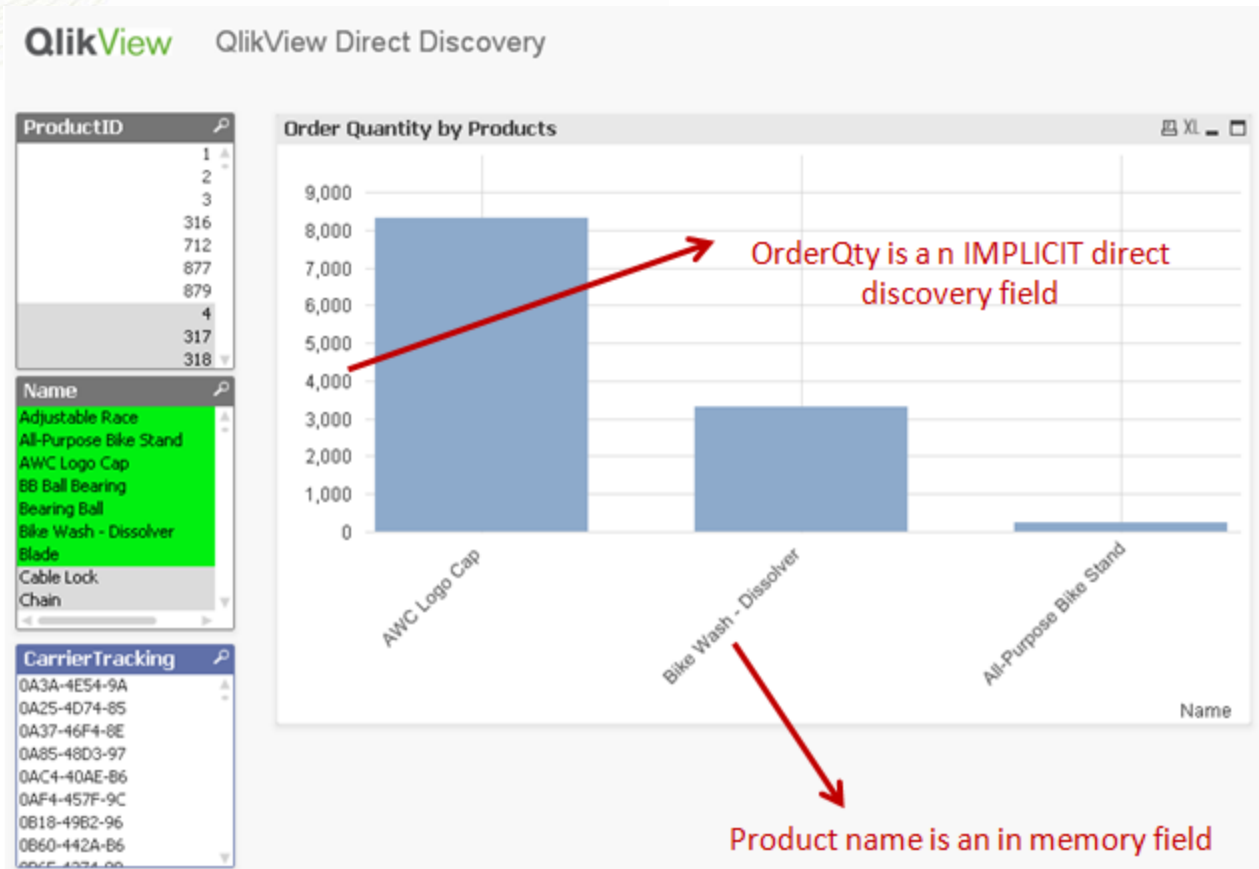


Figure 11. QlikView chart using in-memory and direct discovery fields

Because QlikView is generating SQL statements based on a combination the content of the DIRECT SELECT statement and the content of the chart expressions, some care is required for DBMS column names that require identifier quoting (e.g. column names with spaces or column names that are keywords). As with SQL SELECT statements, the user is responsible for proper column name quoting. In the case of implicit fields that require quoting, the quotes must be applied in the IMPLICIT clause of the DIRECT SELECT statement, not in the chart expression. For databases such as Oracle that control case sensitivity through the use of quoted identifiers, the column names in the chart expressions must match the case of the column names in the database exactly.

Please note that there is a difference in behavior when it comes to handling null values in direct discovery tables. For example, on an in memory table, "SET NullDisplay" will replace all null value occurrences with the string specified, such as: SET NullDisplay='myNullValues'; This is possible

since the in memory table has full access to the whole table in memory. For Direct Discovery tables, it is not possible to process the data in that way since the table is not accessible in memory. So different behaviour will be seen when handling nulls on the direct discovery table. This is the same for using "NullAsValue" and "SET NullValue".

Because of the SQL nature of direct discovery "Suppress missing" option of QlikView charts does not work and should be disabled. Once it is disabled, "Suppress when value is null" option will work as expected (Figure 12).

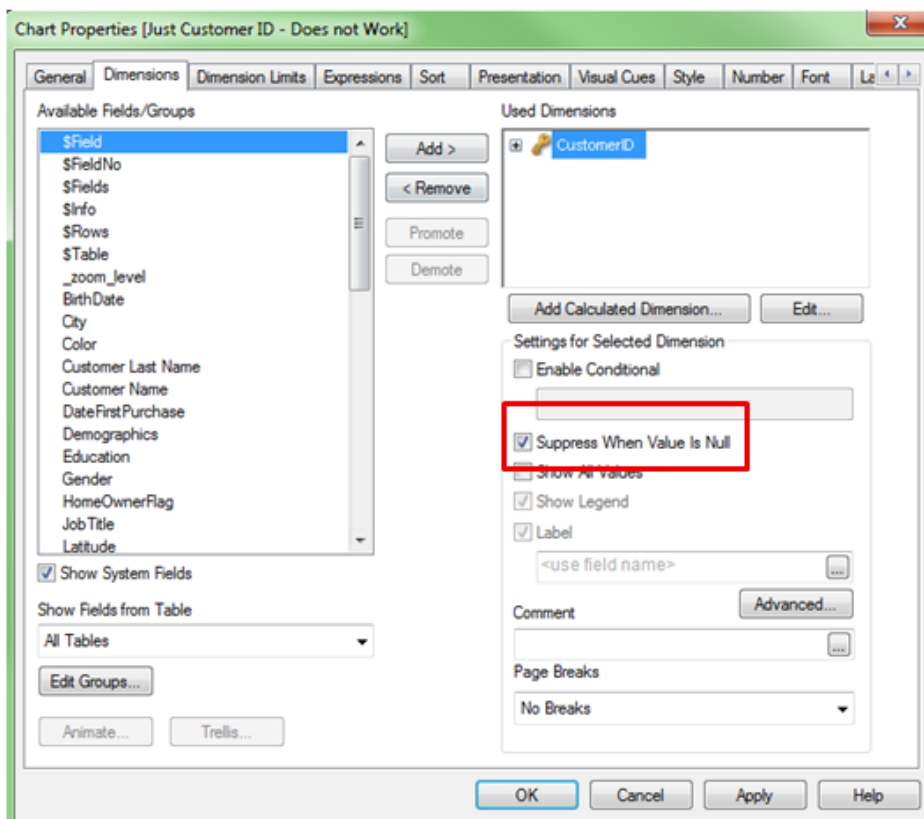


Figure 12. Suppress when value is null option

Supported data types

All data types are supported, however there maybe cases where specific source data formats need to be defined to QlikView, especially working with the date fields. This can be done on the

load script by using the “SET Direct...Format” syntax. Below example demonstrates how to define the date format of the source database that is used as the source for Direct Discovery.

```
SET DirectDateFormat='YYYY-MM-DD';
```

Figure 13. Load Script Syntax to define database specific data formats

There are also 2 new script variables for controlling how direct discovery formats money-type values in the generated SQL statements.

```
SET DirectMoneyFormat (default '#.0000')  
SET DirectMoneyDecimalSep (default '.')
```

The syntax for the two is the same as for MoneyFormat and MoneyDecimalSep, but there are two important differences in the usage:

- This is not a display format, so it should not include currency symbols or thousands separators.
- The default values are not driven by the locale but are hard wired to the values above (the reason for this is that the locale-specific format includes the currency symbol)

Direct Discovery can support the selection of extended Unicode data by using the SQL standard format for extended character string literals (N'<extended string>') as required by some databases (notably SQL Server). The use of this syntax can be enabled for Direct Discovery with the script variable DirectUnicodeStrings. Setting this variable to “true” will enable the use of “N” in front of the string literals.

IV. DATA SOURCES

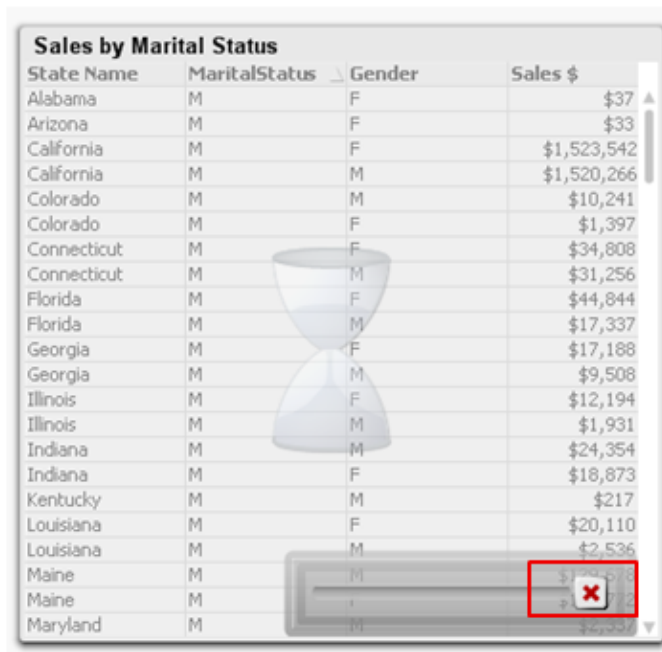
QlikView Direct Discovery can only be used against SQL compliant data sources. The following data sources are supported;

- ODBC/OLEDB data sources - All ODBC/OLEDB sources are supported, including SQL Server, Teradata and Oracle.
- Custom connectors which support SQL – SAP SQL Connector, Custom QVX connectors for SQL compliant data stores.

Both the 32-bit and 64-bit connections are supported with this feature. For the beta release, these are tested data sources: SQL Server 2008, Oracle 11g, Teradata 13.10.

V. CANCELLING DIRECT DISCOVERY QUERY

The cancel icon on QlikView charts can be used to abort the direct discovery query. The direct discovery query running for a chart will be automatically aborted when the QlikView application is closed or a new tab is selected.



The screenshot shows a table with the following data:

State Name	MaritalStatus	Gender	Sales \$
Alabama	M	F	\$37
Arizona	M	F	\$33
California	M	F	\$1,523,542
California	M	M	\$1,520,266
Colorado	M	M	\$10,241
Colorado	M	F	\$1,397
Connecticut	M	F	\$34,808
Connecticut	M	M	\$31,256
Florida	M	F	\$44,844
Florida	M	M	\$17,337
Georgia	M	F	\$17,188
Georgia	M	M	\$9,508
Illinois	M	F	\$12,194
Illinois	M	M	\$1,931
Indiana	M	M	\$24,354
Indiana	M	F	\$18,873
Kentucky	M	M	\$217
Louisiana	M	F	\$20,110
Louisiana	M	M	\$2,536
Maine	M	M	
Maine	M	M	
Maryland	M	M	

A red box highlights a cancel icon (a square with an 'X') located at the bottom right of the table's scroll area.

Figure 14. Cancelling direct discovery query

VI. SUPPORTED QLIKVIEW FUNCTIONALITY

Due to the interactive and SQL syntax specific nature of the Direct Discovery approaches, the following QlikView features are not supported;

- Advanced calculations (Set Analysis, complex expressions)
- Calculated dimensions
- Comparative Analysis (Alternate State) on the QlikView objects that use Direct Discovery fields
- Direct Discovery fields are not supported on Global Search

- Binary load from a QlikView application with Direct Discovery table
- Section access and data reduction
- Loop and Reduce
- Synthetic keys on the Direct Discovery table
- Table naming in script does not apply to the Direct table
- Document chaining
- The use of "*" after DIRECT SELECT keyword on the load script (e.g. DIRECT SELECT *)

VII. PERFORMANCE CONSIDERATIONS

The performance of the Direct Discovery feature fundamentally reflects the performance of the underlying system as the feature queries an external system from QlikView. Please note that, in-memory processing will always be faster than in-database processing. It is possible to use standard database and query tuning best practices for this feature. All of the performance tuning should be done on the source database. Direct Discovery feature does not provide any support for query performance tuning from the QlikView application. However, it is possible to do asynchronous, parallel calls to the database by using the connection pooling capability. The load script syntax to setup the pooling capability is as follows:

```
SET DirectConnectionMax=10;
```

Figure 15. Load Script Syntax to setup the pooling capability

To improve the overall user experience, QlikView's caching is used. Please see the section on caching for more information.

VIII. SECURITY

Some security best practices should be taken into considerations when using Direct Discovery feature.

- All of the users using the same QlikView application with the Direct Discovery capability will be using the same connection. With this initial release, authentication pass through or credentials per user are not supported.
- Section Access is not supported.

- With the new SQL() expression function, it would be possible to execute custom SQL statements in the database. It is advised that the database connection set up in the load script should use an account with only read access to the database.
- Please note that with the use of the new SQL() expression function, it would be possible for the users to create SQL statements that could affect the performance of the database system. If this is the case, the cancel functionality could be used to cancel the queries from the QlikView user interface.
- With this release of QlikView Direct Discovery, there is no logging capability. However it is possible to use the ODBC tracing capability. Please see the logging section for more details on this.
- It is possible to flood the database with requests from the client.
- It is possible to get detailed error messages from the QlikView Server log files.

IX. QLIKVIEW SERVER SETTINGS

Some settings on the QlikView Server and on the config.xml file should be reviewed if the Direct Discovery capability is used on a QlikView application. Please note that once these settings are changed, it will affect all of the QlikView applications that are on the same QlikView Server.

Object Calculation Time Limit

As Direct Discovery feature queries an external system from QlikView, the chart calculation time is dependent on the performance of the underlying system. It is advised to set the object calculation time limit setting on the QlikView Management Console high enough to allow enough time for the QlikView chart to get the direct discovery query results back from the data source. This setting is located under Performance tab of QlikView Server listed on the QlikView Management Console (Figure 16).

Max Symbols in Charts

Max symbols in charts setting is used to set the number of data points to be displayed on QlikView charts. Please note that as Direct Discovery query can return many distinct values, it is advised to review this setting to allow QlikView to display the desired number of data points on charts (Figure 16).

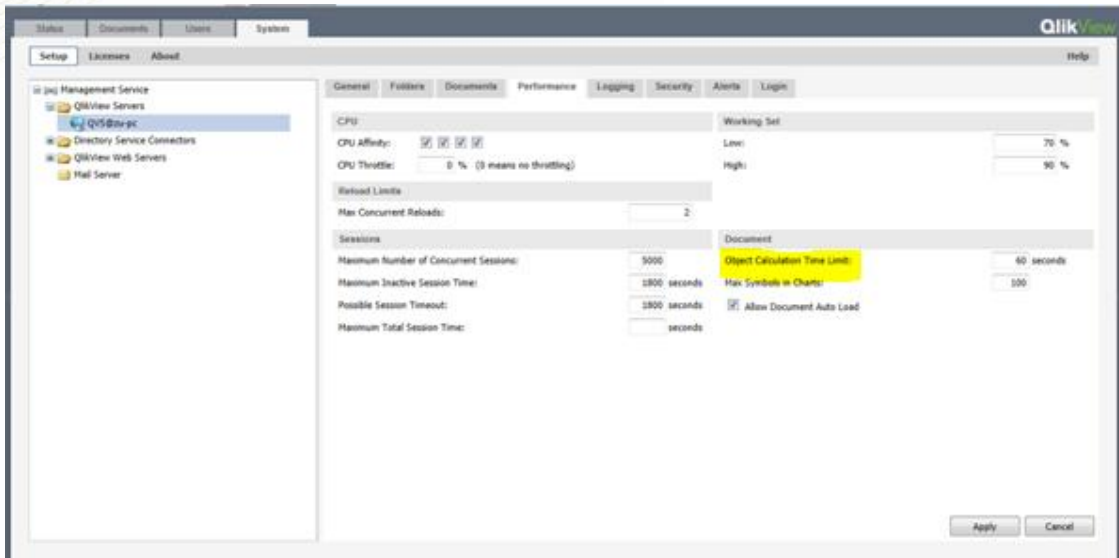


Figure 16. Object calculation time limit and Max symbols in charts settings on QlikView Management Console

QVS Time Out Setting on the Config.xml File

As Direct Discovery feature queries an external system from QlikView, the QlikView Server time out setting on the config.xml file could be adjusted to allow enough time to QlikView to get the query results back. This change should be made if “Lost connection to server” error is seen when using the Ajax client.

The default setting on this option is 60 seconds. It is advised to increment this value to the possible maximum query time. Config.xml file is located under C:\ProgramData\QlikTech\WebServer folder. Please note that during upgrades this file will be overwritten with the default value.

```
<QvsTimeout>60</QvsTimeout>
```

Figure 17. QlikView Server time out setting on config.xml file

X. QLIKVIEW PUBLISHER

When QlikView applications with direct discovery are used with QlikView Publisher, please make sure that the service account running QlikView Publisher has read access on the source direct discovery table. This is required to allow QlikView Publisher access to the direct discovery table during scheduled data refreshes.

XI. CACHING

The performance of querying an external system from QlikView fundamentally reflects the performance of the underlying system. To improve the overall user experience, QlikView’s caching is used and stores selection states of queries in memory. As the same types of selections are made, QlikView will leverage the query from the cache (Figure 19). These cached result sets are shared across users. Please note that, when the same types of selections are done, QlikView will not query the source data and will leverage the cached result set. Also, when back and forward buttons are used, QlikView will leverage the query results from the cache.

It is possible to set a time limit on caching. A special syntax, “STALE” can be used on the load script to set a caching limit on the Direct Discovery query results (Figure 18). Once this time limit hits, QlikView Server will clear the cache for the Direct Discovery query results that were generated for the previous selections. When this happens, QlikView will query the source data for the selections and will create the cache again for the designated time limit.

```

Edit Script [QlikView Direct Discovery]
File Edit Insert Tab Tools Help
Reload Debug
Main
2 SET DecimalSep='.';
3 SET MoneyThousandSep=',';
4 SET MoneyDecimalSep='.';
5 SET MoneyFormat='$#,##0.00; ($#,##0.00)';
6 SET TimeFormat='h:mm:ss TT';
7 SET DateFormat='M/D/YYYY';
8 SET TimestampFormat='M/D/YYYY h:mm:ss[.fff] TT';
9 SET MonthNames='Jan;Feb;Mar;Apr;May;Jun;Jul;Aug;Sep;Oct;Nov;Dec';
10 SET DayNames='Mon;Tue;Wed;Thu;Fri;Sat;Sun';
11
12 ODBC CONNECT TO Test (XUserId is WSMeTzFMbE, XPassword is KOWJCIFMTbcOXXJOELMETeB);
13
14 OrderFactTable:
15 DIRECT (Stale after 5 seconds) SELECT
16     CarrierTrackingNumber
17     ,ProductID
18 FROM AdventureWorks.Sales.SalesOrderDetail ;
19
20 ProductDescription:
21 LOAD
22     [ProductID]
23     ,[Name];
24 SQL SELECT
25     [ProductID]
26     ,[Name]
27 FROM [AdventureWorks].[Production].[Product];
28
  
```

Figure 18. “Stale” option on the load script to clear the direct discovery table cache

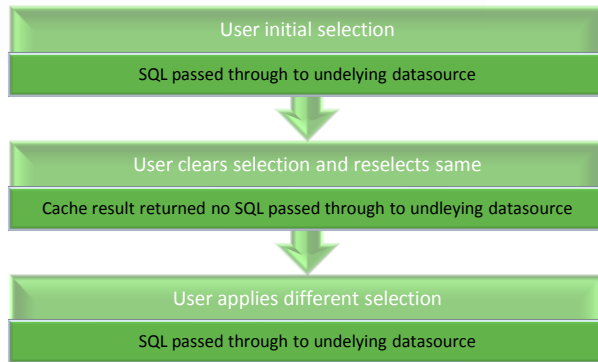


Figure 19. QlikView caching process for Direct Discovery data

The default caching time for direct discovery query results is 30 minutes unless the STALE option is used.

XII. LOGGING

Behind the scenes a direct SQL statement is passed to the underlying data source as such the statement passed can be viewed through the trace files of the underlying connection. An example of this is shown below for a standard ODBC connection:

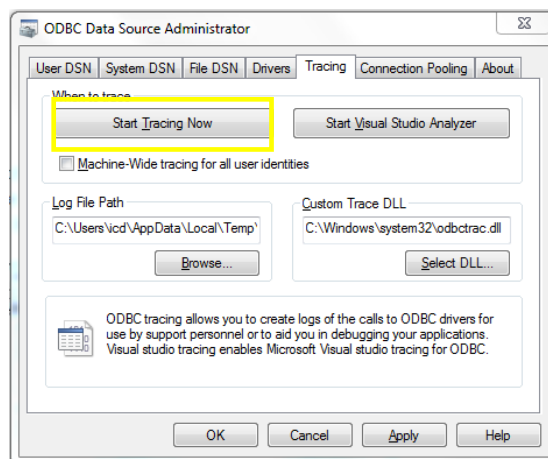
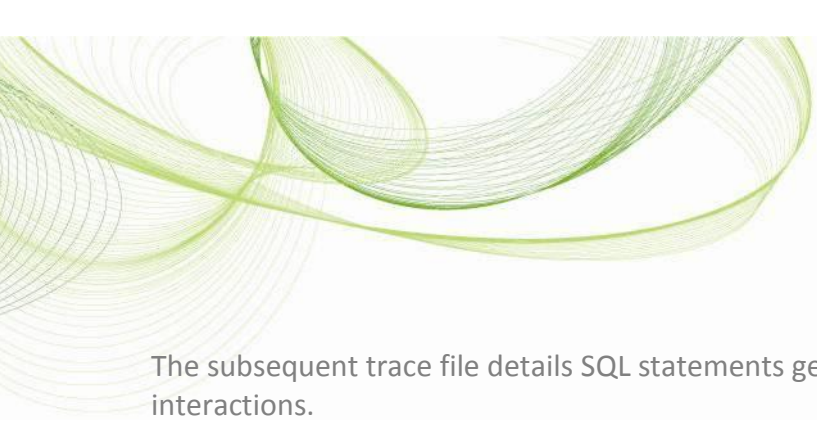


Figure 20. Standard ODBC connection tracing tab

A decorative graphic in the top left corner consisting of several overlapping, wavy, light green lines that create a sense of motion and depth.

The subsequent trace file details SQL statements generated through the user selections and interactions.

XIII. ERROR MESSAGES

Because of the SQL syntax nature of Direct Discovery some errors may happen on the QlikView charts when direct discovery fields are used. Below is a brief description of these errors.

- **Direct query attempted against missing or non-direct table**
This error message is displayed on the chart when the load script has been changed so that a previously defined direct query is now against a table that is no longer there or against an in memory table.
- **Mixed DBMS and in-memory aggregation not supported**
This error message is displayed when the new SQL(“”) expression function is used with snowflake dimension fields.
- **Direct query failed**
This error message is displayed when the query that is executed for direct discovery is failed.

XIV. KNOWN ISSUES

Please refer to QlikView 11.2 release notes document on QlikView download site.

XV. ADDITIONAL RESOURCES

QlikView Direct Discovery Data Sheet

<http://www.qlikview.com/us/explore/resources/brochures-datasheets?language=english>

QlikView Direct Discovery FAQ

<http://www.qlikview.com/us/explore/resources/brochures-datasheets?language=english>

QlikView and Big Data

<http://www.qlikview.com/us/explore/products/big-data>