



QlikView 11.2 SR5 DIRECT DISCOVERY

QlikView Technical Addendum

Published: November, 2012

Version: 5.0

Last Updated: December, 2013

www.qlikview.com

Contents

1	<i>Overview</i>	3
2	<i>What is QlikView Direct Discovery</i>	3
3	<i>Technical Details</i>	4
3.1	How Does Direct Discovery Work?	4
3.2	Data Loading	5
3.3	Creating QlikView Objects with Direct Discovery Fields	11
3.4	Supported data types and set statements	15
3.5	Data Sources	17
3.6	Cancelling a Direct Discovery Query	18
3.7	Unsupported QlikView Functionality	18
4	<i>Performance Considerations</i>	19
5	<i>Security</i>	19
6	<i>QlikView Server Settings</i>	20
6.1	QlikView Publisher	21
6.2	Caching	21
6.3	Error Messages	22
6.4	Logging	23
7	<i>Under The Hood</i>	24
7.1	In Memory	24
7.2	Direct Discovery	25

1 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 larger data sources. It provides QlikView's complete associative experience on top of data coming directly from external larger 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.

2 What is QlikView Direct Discovery

QlikView Direct Discovery capability combines the associative capabilities of the QlikView in memory dataset with a query model where not all of the source data is 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, grey, and white. They can create charts that would help them analyse data from both data sets together.

3 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.

3.1 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 QUERY". 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 in 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 in 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.

3.2 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:

```
ODBC CONNECT TO AdWorks;

LOAD CustomerID,
SalesPersonID,
SalesOrderID,
OrderDate,
month([OrderDate]) as OrderMonth,
year([OrderDate]) as OrderYear,
SubTotal,
TaxAmt,
TotalDue
DueDate,
ShipDate,
AccountNumber,
CreditCardApprovalCode,
rowguid,
ModifiedDate
SQL SELECT
CustomerID,
SalesPersonID,
SalesOrderID,
OrderDate,
SubTotal,
TaxAmt,
TotalDue
RevisionNumber,
DueDate,
ShipDate,
AccountNumber,
CreditCardApprovalCode,
rowguid,
ModifiedDate
FROM AdventureWorks.Sales.SalesOrderHeader;
```

Figure 1. Traditional QlikView Load Script Syntax

To invoke the direct discovery method, the keyword “SQL SELECT” is replaced with “DIRECT QUERY” and additional keywords – DIMENSION, MEASURE, DETAIL and NATIVE are introduced:

```
ODBC CONNECT TO AdWorks;

DIRECT QUERY
  DIMENSION
    CustomerID,
    SalesPersonID,
    SalesOrderID,
    OrderDate,
    NATIVE('month([OrderDate])') as OrderMonth,
    NATIVE('Year([OrderDate])') as OrderYear
  MEASURE
    SubTotal,
    TaxAmt,
    TotalDue
  DETAIL
    DueDate,
    ShipDate,
    AccountNumber,
    CreditCardApprovalCode,
    rowguid,
    ModifiedDate
FROM AdventureWorks.Sales.SalesOrderHeader;
```

Figure 2. QlikView Load Script Syntax for Direct Discovery

In the example above, the source data table “SalesOrderHeader” has 16 fields, with the use of “DIRECT QUERY” and “DIMENSION” keywords, only columns CustomerID, SalesPersonID, SalesOrderID and OrderDate are loaded into memory as symbol tables. Other columns following the “MEASURE” and “DETAIL” keywords exist in the source data table within the database and they are not part of the in memory data model. Both measure and detail fields are fields that QlikView is aware of on a “meta level”. The actual data of measure/detail fields resides only in the database but the fields may be used in QlikView expressions. More information on the measure/detail fields are 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.

The script would produce the following SQL passed through to the source database:

```
SELECT DISTINCT "CustomerID" FROM "AdventureWorks"."Sales"."SalesOrderHeader
```

```

SELECT DISTINCT "SalesPersonID" FROM "AdventureWorks"."Sales"."SalesOrderHeader"
SELECT DISTINCT "SalesOrderID" FROM "AdventureWorks"."Sales"."SalesOrderHeader"
SELECT DISTINCT "OrderDate" FROM "AdventureWorks"."Sales"."SalesOrderHeader"
SELECT DISTINCT month([OrderDate]) FROM "AdventureWorks"."Sales"."SalesOrderHeader"
SELECT DISTINCT year([OrderDate]) FROM "AdventureWorks"."Sales"."SalesOrderHeader"

```

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

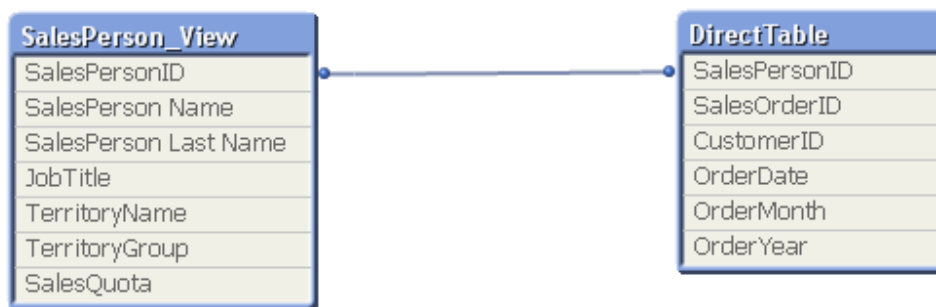


Figure 3. Associative Data Model for Direct Discovery and In-memory data sets

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, "SalesPerson_View" table is loaded in memory and joined to the direct discovery data with the SalesPersonID 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 colours; green, grey, and white.

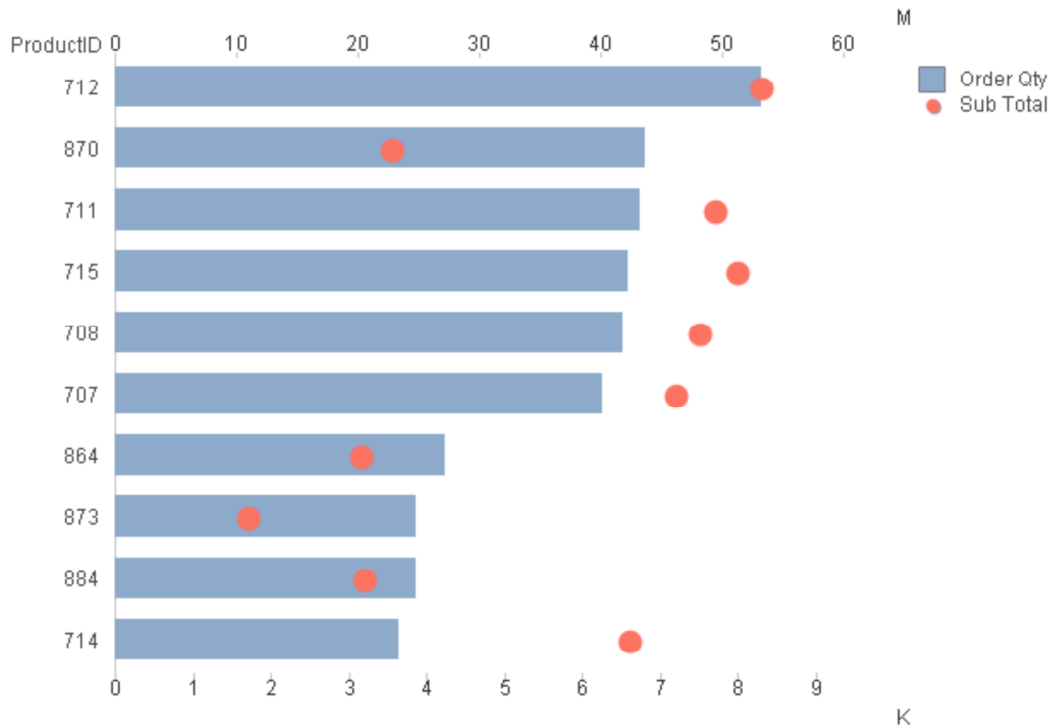


Figure 4. Associative Business Discovery on direct discovery and in-memory data sets

It is also possible to execute source Database SQL functions with the direct discovery table on the load script by using the keyword NATIVE which should be used within single quotation marks.

```
NATIVE ('month([OrderDate])') as OrderMonth,
NATIVE ('Year([OrderDate])') as OrderYear
```

Statements containing single quotes will require additional quotes:

```
NATIVE ('CAST(DAY(SYSDATETIME()) AS VARCHAR(2)) + '/' + CAST(MONTH(SYSDATETIME()) AS VARCHAR(2)) + '/' + RIGHT(CAST(YEAR(SYSDATETIME()) AS VARCHAR(4)), 2) as [D/M/YY]
```

Figure 5. Use of the NATIVE function with the direct discovery table in the load script

This version of direct discovery only supports one direct discovery table per QlikView application, the source of the direct discovery can be one database table or one view. It is also possible to use a WHERE statement on the load script with the direct discovery table.

3.2.1 Background and definition of measure fields

A measure field is a field that QlikView is aware of on a “meta level”. When QlikView recognizes the keyword “DIRECT QUERY” in conjunction with “MEASURE” on the load script, it only loads the metadata of the fields from the source table. The actual data of a measure field resides only in the database but the field may be used in QlikView expressions. The idea is that a measure field will be treated as any other field when the user works with expressions in QlikView. The measure fields are required to have an aggregation function when used on the user interface. When QlikView generates queries including measure direct discovery fields, it uses a GROUP BY on the SQL statement with the corresponding dimension.

The reason to introduce the concept of measure 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 Sum(SubTotal) is used in a QlikView chart and SubTotal is a measure field QlikView will let the database do the sum. Measure fields will add support for aggregating in the database while using the same expression syntax. The result is that the usage of the direct discovery fields from the database will be transparent for the user.

3.2.2 Dimension direct discovery fields

Dimension direct discovery fields are the fields that are listed after DIRECT QUERY and DIMENSION keywords in the load script. The unique data values of these fields are loaded into the in memory symbol tables. The main use for dimension fields are;

- To define dimension chart values
- To create the association SQL between the in-memory data and the direct discovery data
- To define list boxes with direct discovery data

Please note that when the dimension direct discovery fields are used in list boxes, the data values displayed in the list boxes will not get updated UNLESS the application is reloaded.

3.2.3 Detail direct discovery fields

Drill to details is a new capability introduced in 11.2 SR5 and allows “detail” records to be displayed in a QlikView table box only. A detail record is one to which no aggregation is applied and effectively just generates a select statement from QlikView and renders the appropriate values. To cater for the potential large volume of data which could be rendered an additional Set variable is introduced:

```
SET DirectTableBoxListThreshold = 100000;
```

The default value is 1000.

As part of the drill to details functionality additional syntax has been introduced this follows the DIRECT QUERY statement:

```
DIRECT QUERY
DIMENSION
  CustomerID,
  SalesPersonID,
  SalesOrderID,
  OrderDate,
  NATIVE('month([OrderDate])') as OrderMonth,
  NATIVE('Year([OrderDate])') as OrderYear
MEASURE
  SubTotal,
  TaxAmt,
  TotalDue
DETAIL
  DueDate,
  ShipDate,
  AccountNumber,
  CreditCardApprovalCode,
  rowguid,
  ModifiedDate
FROM AdventureWorks.Sales.SalesOrderHeader;
```

Figure 6. Use of DETAIL syntax in the load script

DETAIL - These are the fields that the user may want to display in a drill-to-details table box but that should not be involved in any chart expressions. The data will be displayed at the lowest level without aggregation within a table box.

3.2.4 Detach direct discovery fields

The Detach keyword has been introduced to flag certain dimension fields NOT to be part of the associative experience BUT to be part of the query generated passed to the data source. The main use case for this is for large volume granular dimension values to increase the chart rendering time due to bypassing the associative capability, section 7 describes this in more detail.

As part of the DETACH functionality additional syntax has been introduced to flag the dimensions to be used in this mode:

```

DIRECT QUERY
DIMENSION
    SalesPersonID,
    OrderDate,
    NATIVE('month([OrderDate])') as OrderMonth,
    NATIVE('Year([OrderDate])') as OrderYear
MEASURE
    SubTotal,
    TaxAmt,
    TotalDue
DETAIL
    DueDate,
    ShipDate,
    AccountNumber,
    CreditCardApprovalCode,
    rowguid,
    ModifiedDate
DETACH
    SalesOrderID,
    CustomerID
FROM AdventureWorks.Sales.SalesOrderHeader;

```

Figure 7. Use of DETACH syntax in the load script

In the above example SalesOrderID and CustomerID would appear greyed out in a list box and not be part of the associative experience but will form part of the WHERE clause passed to the underlying data source if selected in a list box.

3.3 Creating QlikView Objects with Direct Discovery Fields

Due to the interactive and SQL specific nature of QlikView Direct Discovery, only certain QlikView objects can use measure direct discovery fields. Statistics boxes are not supported with measure direct discovery fields.

On the field list, additional information is provided to notify the user that the field is a measure field (figure 8). This is useful information as some of the measure fields from the big data sources may have a 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 measure fields. Although the data values for the measure fields are not loaded in memory, they still consume memory and CPU once they are used on the user interface of a QlikView application.

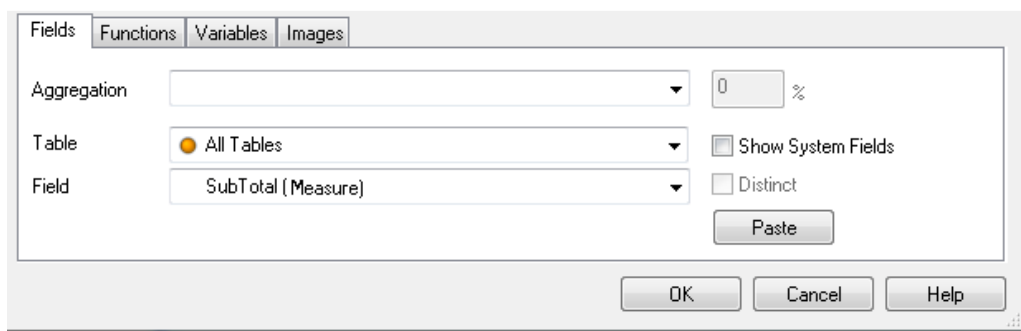


Figure 8. The direct discovery fields are marked with the keyword “Measure” on the field list.

3.3.1 How to create list boxes with direct discovery fields

The traditional way of creating list box is used to create list boxes with the dimension direct discovery fields (fields that are listed after “DIRECT QUERY DIMENSION” keywords in the load script).

In the previous example CustomerID, SalesPersonID, SalesOrderID , OrderDate, OrderMonth and OrderYear fields can be used in list boxes as these are the dimension 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.

It is possible to use the expression option of the list box with a measure direct discovery field. The `aggr()` function should be used to show the aggregated value of the measure field for any dimension field. For example, the list box expression “`aggr (sum (SubTotal) , CustomerID)`” will list the aggregated SubTotal values for CustomerId names in a list box.

3.3.2 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 measure direct discovery fields can be used in expressions and dimension direct discovery fields can be used as dimensions. In the previous example, only CustomerID, SalesPersonID, SalesOrderID, OrderDate, OrderMonth and OrderYear fields can be used as chart dimensions and SubTotal, TaxAmt and TotalDue 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.

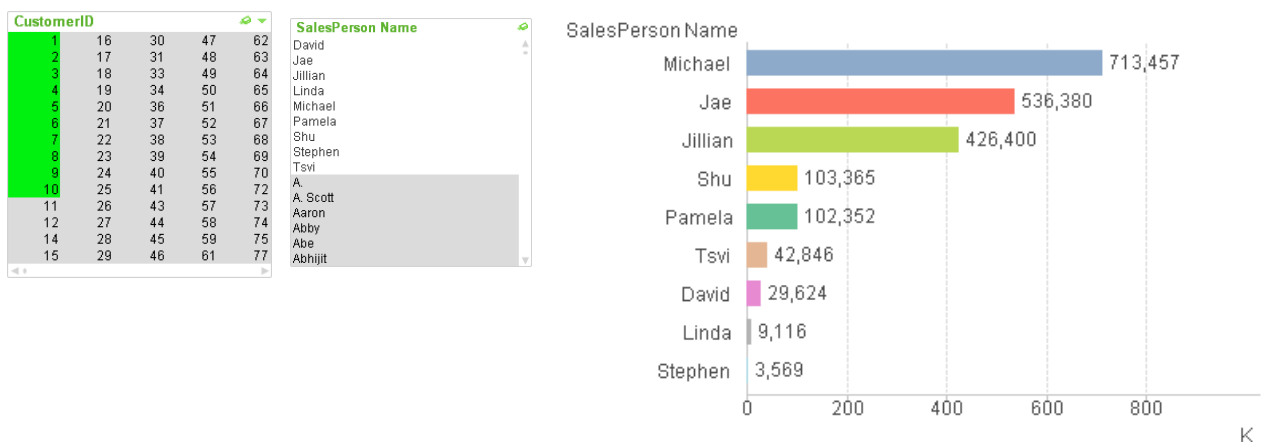


Figure 9. QlikView chart/selections using direct discovery fields as dimension and expression

The sheet in Figure 9 would produce the following SQL pushed down to the database:

Associations are NOT stored in memory and are built from the source:

```
SELECT DISTINCT "SalesPersonID" FROM "AdventureWorks"."Sales"."SalesOrderHeader" WHERE "CustomerID" IN ( 1, 2, 3, 4, 5, 6, 7, 8, 9, 10 )
```

And for the chart:

```
SELECT "SalesPersonID", SUM("SubTotal") FROM "AdventureWorks"."Sales"."SalesOrderHeader"  
WHERE "CustomerID" IN ( 1, 2, 3, 4, 5, 6, 7, 8, 9, 10 ) GROUP BY "SalesPersonID"
```

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. 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 ...).

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.

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.

3.3.3 How to display drill to detail records

Detail records can ONLY be displayed in a Table box, the Table box MUST contain a DIMENSION field (CustomerID) and can also contain both DETAIL (CreditApprovalCode, AccountNumber, DueDate) and MEASURE (SubTotal, TaxAmt) fields.

CustomerID	AccountNumber	DueDate	CreditCardApprovalCode	SubTotal	TaxAmt
1	10-4020-000001	13/02/2002 12:00:00 AM	105493VI99678	£34,066.19	£2,725.30
1	10-4020-000001	13/08/2001 12:00:00 AM	105571VI99678	£13,216.05	£1,057.28
1	10-4020-000001	13/11/2001 12:00:00 AM	105516VI99678	£23,646.03	£1,891.68
1	10-4020-000001	13/05/2002 12:00:00 AM	105533VI99678	£31,423.52	£2,513.88
2	10-4020-000002	13/02/2004 12:00:00 AM	17172VI51504	£3,668.73	£293.50
2	10-4020-000002	13/11/2002 12:00:00 AM	15790VI51504	£4,949.86	£395.99
2	10-4020-000002	13/08/2002 12:00:00 AM	15832VI51504	£9,216.36	£737.31
2	10-4020-000002	13/05/2004 12:00:00 AM	17000VI51504	£822.01	£65.76
2	10-4020-000002	13/11/2003 12:00:00 AM	15802VI51504	£4,106.65	£328.53
2	10-4020-000002	13/08/2003 12:00:00 AM	15862VI51504	£3,534.17	£282.73
2	10-4020-000002	13/02/2003 12:00:00 AM	15786VI51504	£1,574.12	£125.93
2	10-4020-000002	13/05/2003 12:00:00 AM	15795VI51504	£1,751.60	£140.13
3	10-4020-000003	13/03/2004 12:00:00 AM	38415VI26986	£15,431.52	£1,234.52
3	10-4020-000003	13/12/2001 12:00:00 AM	38408VI26986	£17,419.61	£1,393.57
3	10-4020-000003	13/03/2003 12:00:00 AM	38398VI26986	£27,899.96	£2,232.00
3	10-4020-000003	13/12/2002 12:00:00 AM	38392VI26986	£51,198.54	£4,095.88
3	10-4020-000003	13/06/2003 12:00:00 AM	38428VI26986	£47,138.00	£3,771.04
3	10-4020-000003	13/06/2002 12:00:00 AM	38384VI26986	£11,796.18	£943.69
3	10-4020-000003	13/09/2003 12:00:00 AM	38442VI26986	£83,436.18	£6,674.89
3	10-4020-000003	13/09/2002 12:00:00 AM	38431VI26986	£71,112.13	£5,688.97
3	10-4020-000003	13/06/2004 12:00:00 AM	37038VI26986	£31,559.75	£2,524.78
3	10-4020-000003	13/03/2002 12:00:00 AM	38389VI26986	£18,285.64	£1,462.85
3	10-4020-000003	13/12/2003 12:00:00 AM	37443VI26986	£40,109.06	£3,208.72
3	10-4020-000003	13/09/2001 12:00:00 AM	38409VI26986	£18,555.81	£1,484.47
4	10-4020-000004	13/07/2003 12:00:00 AM	96519VI75503	£73,368.09	£5,869.45

Figure 10. Drill to details table box

Potentially a large amount of data could be retrieved from this process as such the table box will not display any data until the following SQL is executed:

```
SELECT COUNT(*) FROM "AdventureWorks"."Sales"."SalesOrderHeader"
```

If the result of the count(*) is under the threshold of the following set variable

```
SET DirectTableBoxListThreshold = 100000;
```

From the script then the data will be retrieved from the source database with the following SQL:

```
SELECT "CustomerID", "CreditCardApprovalCode", "AccountNumber", "DueDate", "SubTotal", "TaxAmt" FROM "AdventureWorks"."Sales"."SalesOrderHeader"
```

3.4 Supported data types and set statements

All data types are supported, however there may be 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..." syntax. Below example demonstrates how to define the date format of the source database that is used as the source for Direct Discovery.

Set Statement	Description
DirectDateFormat	Controls the format of the date literals sent to the database
DirectTimestampFormat	Controls the format of the datetime literals sent to the database
DirectTimeFormat	Controls the format of the time literals sent to the database
DirectUnicodeStrings	Controls whether string literals are prepended with the ANSI standard wide character marker N. Not all databases support this. Set to TRUE to enable; any other setting disables.
DirectMoneyFormat	Controls the format of the money literals sent to the database, This is not a display format, so it should not include currency symbols or thousands separators, default '#.0000'
DirectMoneyDecimalSep	Controls the decimal separator used in money literals sent to the database. Note that this character must match the character used in DirectMoneyFormat . Default is '.'
DirectStringQuoteChar	Controls the character to be used to quote the strings in a generated query. The default is a single quote.
DirectIdentifierQuoteChar	Controls the quoting of identifiers in a generated query. This can be set to either one character (such as a double quote) or two (such as the pair "[]"). The default is a double quote.
DirectCacheSeconds	Controls the amount of time a query result set from an application is stored in memory cache, default setting is 3600 seconds.
DirectTableBoxListThreshold	Controls the amount of drill to detail data available to be shown in a table box, the default is 1000 rows
DirectConnectionMax	Controls it the number of parallel calls to the database by using the connection pooling capability, the default is 1
DirectDistinctSupport	Changes the default syntax of SELECT DISTINCT to use GROUP BY for sources which do not support DISTINCT
DirectIdentifierQuoteStyle	For non-standard ANSI quoting anything other than 'ANSI' in the statement will change the syntax

3.5 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.

3.5.1 SAP as a Data Source

Direct discovery can be used in conjunction with the QlikView SAP SQL Connector only and requires the following parameters in the set variables:

```
SET DirectFieldColumnDelimiter=' ';  
SET DirectIdentifierQuoteChar=' ';
```

SAP uses OpenSQL which delimits columns with a space rather than a comma so the above set statements will cater for this difference to ANSI SQL.

3.5.2 Google Big Query as a Data Source

Direct discovery can be used in conjunction with Google Big Query and requires the following parameters in the set variables:

```
SET DirectDistinctSupport=false;  
SET DirectIdentifierQuoteChar='[]';  
SET DirectIdentifierQuoteStyle='big query';
```

Google Big Query does not support Select distinct or quoted column/table names and has non ANSI quoting configuration using '[]'.

3.5.3 MySQL and MS Access as a Data Source

Direct discovery can be used in conjunction with MySQL and MS Access but may require the following parameters in the set variables due to the quoting characters used in these sources:

```
SET DirectIdentifierQuoteChar='`';
```

3.6 Cancelling a 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.

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	F	
Maryland	M	M	

Figure 11. Cancelling direct discovery query

3.7 Unsupported QlikView Functionality

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

- Set Analysis
- Calculated dimensions
- Direct Discovery Measure and Detail fields are not supported on Global Search
- 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

4 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;
```

The default value is set to 1.

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

5 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 NATIVE() 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.
- 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.

6 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 12).

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 12).

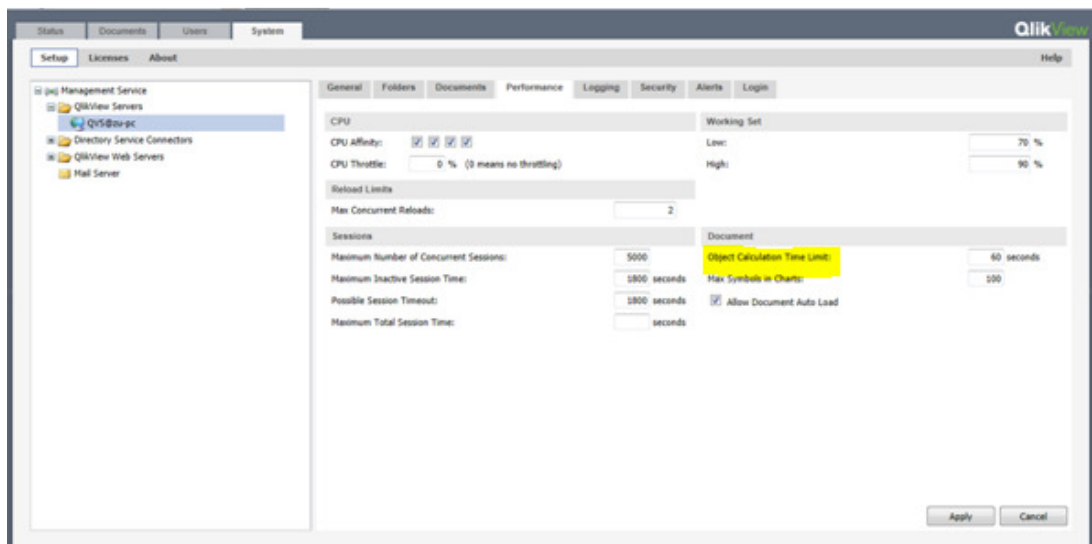


Figure 12. Object calculation time limit and Max symbols in charts settings on QlikView Management Console

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>
```

6.1 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.

6.2 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 13). 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 via a set variable:

```
SET DirectCacheSeconds= 10;
```

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.

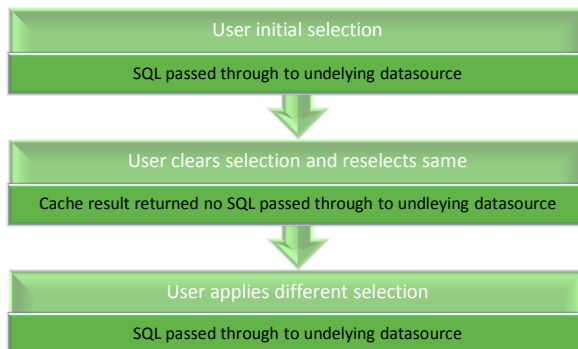


Figure 13. QlikView caching process for Direct Discovery data

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

6.3 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.
- Direct query failed
 - This error message is displayed when the query that is executed for direct discovery is failed and could be caused by a number of factors, see section 6.4 to activate logging to investigate further.
- Table box row limit exceeded
 - This warning is specific to the drill to details feature for a table box where the amount of rows to be retrieved is over the SET TableBoxThresholdListThreshold

parameter (default is 1000).

- Database connection failed
 - This error message is displayed during chart execution (data is not cached) when a communication error between QlikView and the source database occurs.

6.4 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:

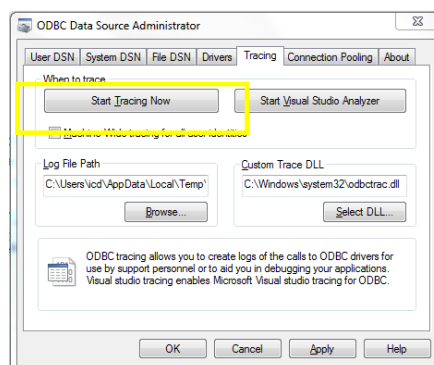


Figure 14. Standard ODBC connection tracing tab

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

Logging can provide a more detailed explanation of the generic “Direct Query Failed” message, for example building a chart with the following expression

`TextCount(CustomerID)`

would result in the chart failing to render and write the following to the log file:

[Microsoft][ODBC SQL Server Driver][SQL Server]'TEXTCOUNT' is not a recognized built-in function name.

7 Under The Hood

To understand the differences between the behaviour of an in memory versus a Direct Discovery load script its necessary to look at how QlikView populates its internal structures during both of these processes

7.1 In Memory

Taking the following script as an example:

```
LOAD CustomerID, SalesPersonID, SalesOrderID, SubTotal
SQL SELECT CustomerID, SalesPersonID, SalesOrderID, SubTotal
FROM AdventureWorks.Sales.SalesOrderHeader;
```

During the reload process the data is transformed into two table types one associative data table and multiple symbol tables (one per field). The symbol tables contain one row per distinct value of the field. Each row contains a pointer and the value of the field, both the numeric value and the textual component. Basically, the symbol tables are look-up tables for the field values as shown in figure 15.

Unique values are loaded into the structures (symbol tables)

<u>CustomerID</u>		<u>SalesPersonID</u>		<u>SalesOrderID</u>		<u>SubTotal</u>	
Pointer	Value	Pointer	Value	Pointer	Value	Pointer	Value
10	Acme Ltd	100	Paul S.	1101	AB1234	10001	2345.57
11	Smith Co.	101	Mary D.	1110	AC1234	10010	45.78

The associative data table is simultaneously populated

<u>CustomerID</u>	<u>SalesPersonID</u>	<u>SalesOrderID</u>	<u>SubTotal</u>
10	100	1101	10001
11	100	1110	10010
10	101	1110	10001
10	100	1101	10010
...


Figure 15. In-memory data load

7.2 Direct Discovery

Modifying the script to enable Direct Discovery:

```
DIRECT QUERY DIMENSION CustomerID, SalesPersonID, SalesOrderID
MEASURE SubTotal
FROM AdventureWorks.Sales.SalesOrderHeader;
```

During the reload process the only the DIMENSION data is transformed into multiple symbol tables (DETAIL and MEASURE column data remains in the source) and the associative data table remains empty with only the metadata being loaded (column names) as shown in figure 16.

<u>CustomerID</u>		<u>SalesPersonID</u>		<u>SalesOrderID</u>		<u>SubTotal</u>
Pointer	Value	Pointer	Value	Pointer	Value	
10	Acme Ltd	100	Paul S.	1101	AB1234	
11	Smith Co.	101	Mary D.	1110	AC1234	

The associative data table is not populated on reload and created by selections (with caching)

<u>CustomerID</u>	<u>SalesPersonID</u>	<u>SalesOrderID</u>	<u>SubTotal</u>



Figure 16. Direct discovery data load

The association table is populated on the fly per user click with the result set being cached. The exception to this rule would be dimension fields flagged as “DETACH” which would not populate the association data table at all resulting in a faster query execution BUT without any associations.