IBM Business Analytics Proven Practices: The Cognos BI 10.1.1 Dynamic Query Cookbook

Product(s): IBM Cognos BI 10.1.1; Area of Interest: Infrastructure

Daniel Wagemann
Armin Kamal
Pierre Valiquette
Rick Kenny
Tod Creasey

May 13, 2013
(First published October 27, 2010)

This document is intended to provide a single point of reference for techniques and product behaviours when dealing with the Dynamic Query Mode delivered with IBM Cognos 10. For additional resource material, please see the IBM Cognos Dynamic Query RedBook.

View more content in this series

Introduction

Purpose

This document provides a single point of reference for techniques and product behaviors when dealing with the Dynamic Query Mode delivered with IBM Cognos 10.

Applicability

The techniques and product behaviors outlined in this document apply to

• IBM Cognos Business Intelligence 10.1.1 build 6235.144

The IBM Cognos Business Intelligence 10.1 PDF version of this document is available within the Downloads section at the bottom of this document.

Exclusions and Exceptions

The techniques and product behaviors outlined in this document may not be applicable to future releases.
Related Documents

For information on making the transition from the Compatible Query Mode to the Dynamic Query Mode in IBM Cognos 10 Business Intelligence, refer to the IBM Cognos 10 Dynamic Query Mode Migration Scenarios on IBM developerWorks at http://www.ibm.com/developerworks/data/library/cognos/upgrade_and_migration/bi/page568.html.


Notation

This document uses the following conventions in filename patterns:

- Italics indicate a portion of a name that will be replaced with an appropriate string for each instance of such a file. For example, logs/XQE/reportName/planningLog.xml is a pattern for files with pathnames such as logs/XQE/QuarterlySalesReport/planningLog.xml.
- c10\ in italics at the beginning of a file path denote the installation directory. For example, c10\bin\ refers to the bin directory under the installation directory.

Overview of the IBM Cognos 10 Dynamic Query Mode

The Dynamic Query Mode is an enhanced Java-based query mode which offers the following key capabilities:

- Query optimizations to address query complexity, data volumes and timeliness expectations with improved query execution techniques
- Significant improvement for complex OLAP queries through intelligent combination of local and remote processing and better MDX generation
- Support for relational databases through JDBC connectivity
- OLAP functionality for relational data sources when using a dimensionally modeled relational (DMR) package
- Security-aware caching
- New data interfaces leveraging 64-bit processing
- Ease of maintenance with query visualization

Query Optimization

The optimization of the queries is achieved through the advanced application of strict query planning rules. These planning rules incorporate the next generation planning approach, which is more streamlined and produces higher quality queries that are faster to execute. The query planning process is also optimized to make better use of metadata and expression level caches, including plan caches which provide higher application throughput.

Performance Improvement through Balanced Local Processing Facilities

The Dynamic Query Mode makes intelligent, rules based and system load based decisions on which parts of a query should be executed locally in the application server versus remotely in the
database server. This ensures that users have the highest functionality possible regardless of whether the underlying data source supports the business intelligence report intent. In addition the Dynamic Query Mode contains a fine grained metadata and cell data cache which is trickle fed and a higher cache hit ratio than was previously possible. In addition the queries which are sent to remote data sources are further optimized by the execution layer based on cache content and advanced null suppression logic.

**OLAP Functionality for Relational Data Sources**

The Dynamic Query Mode offers a true OLAP-over-relational experience when using a dimensionally modeled relational (DMR) package. Dimensional modeling of relational data enables OLAP presentation of metadata, drill up and drill down functionality, and the use of OLAP functions. OLAP data representation enables simple navigation, clear data context, and enhanced data visualization. The Dynamic Query Mode leverages a properly built dimensional layer to deliver consistent OLAP style reporting. It is highly recommended that the dimensional layer be constructed on a relational layer that applies star schema concepts. In Dynamic Query Mode, the OLAP-over-relational technology can be used with dimensional objects only (regular and measure dimensions). This dimensional layer provides an abstraction from relational objects (i.e. data source query subjects) and functions enabling consistent OLAP behavior. Utilizing OLAP over relational with the Dynamic Query Mode will ensure that list and crosstab reports return identical results.

The Dynamic Query Mode applies advanced OLAP caching techniques to enhance performance of dimensionally modeled relational packages. The use of caching reduces the frequency of database queries, thus minimizing the database server workload required to service the IBM Cognos application.

**Security-Aware Caching**

The caching logic available in Dynamic Query Mode is able, when connected to secured metadata sources, to determine the secured access capabilities of each user as they access the data source. This information is then used to optimize the memory usage and internal representation of that user’s secured view of the data source metadata. Security can also be setup so that entire OLAP dimensions can be shared providing cache reuse and performance gains.

**New Data Interfaces Leveraging 64 bit Processing**

The Dynamic Query Mode is a fully 64 bit capable environment for data access. It permits the use of 64 bit data source drivers and can leverage the 64 bit address space for query processing, metadata caching and data caching.

**Ease of Maintenance with Query Visualization**

Query visualization allows system administrators to analyze the queries generated by the Dynamic Query Mode and understand how they will be processed. These visualizations include cost based information derived from the query execution. This information permits the rapid identification of model and query optimizations which could be applied in order to achieve better performance.
The IBM Cognos 10 Architecture

Architecture of Dynamic Query Mode

The Dynamic Query Mode server accepts data and metadata requests (via the Report Service server) from BI studios (such as Report Studio), the Report Viewer and other clients. It returns the requested data and or messages in a structured response to the Report Service, which formats the result for the client.

The following diagram shows the internal architecture of the Dynamic Query Mode server, which consists of the following major components:

- Transformation Engine and Transformation Libraries
- Query Execution Engine
- Member Cache
- Data Cache
- RDBMS and OLAP Adapters

Figure 1 Internal architecture of the Dynamic Query Mode server

The diagram also portrays a variety of relational and dimensional data providers. For a complete and up-to-date listing of supported data sources and their usage, refer to the conformance page located at http://www.ibm.com/support/docview.wss?uid=swg27014782.

The Transformation Engine does not implement any query planning logic by itself. Instead, it provides an execution environment for query transformations located in the Transformation
Libraries, thus separating planning logic from the engine. The transformations implement query planning logic for all supported query types and functionality. When there are no more transformations to be applied, query planning is complete and the Transformation Engine passes the resulting run tree to the Query Execution Engine.

The Query Execution Engine can execute any query request, independent of the type of query and target data source. The engine represents all query results in memory in a single format encompassing both dimensional style (with axes, dimensions and cells) and relational style (tabular format with rows and columns). This allows it to combine SQL and MDX queries in a single run tree, thus enabling simplicity of representation, flexibility in post-processing and streamlined query performance. In order to process the two types of queries, it contains both SQL and MDX engines.

The SQL engine obtains data directly from the RDBMS Adapter. The Query Execution Engine updates the secure Data Cache with dimensional data for future reuse. The MDX engine obtains dimensional data either directly from the OLAP Adapters or from the Data Cache. It also updates and reuses dimensional metadata in the secure Member Cache. The cache security features ensure that no sharing of secured data ever occurs.

The RDBMS and OLAP Adapters translate IBM Cognos SQL and MDX queries to a query dialect suitable for each data provider. They send the query and fetch results through the provider’s proprietary interface or a supported standard interface such as ODBC or JDBC. There is only one RDBMS Adapter, which uses a JDBC interface, because all supported relational providers are accessible through JDBC. The RDBMS Adapter supplies data to the SQL engine in the Query Execution Engine and the OLAP Adapters supply data to the MDX engine.

Planning and Executing the Query

The Dynamic Query Mode has two major components involved in the processing of requests: the Transformation Engine and the Query Execution Engine. Both engines share a common environment and operate on the same query structures: the Plan tree and the Run tree.

An XML parser converts an incoming report request into an initial plan tree, including any embedded SQL or MDX queries. The tree has two main branches: the Query, describing what the user wants to see, and the QueryResultSet, describing how the user wants to see the results (list or crosstab).

With the tree in place, the planning process can begin. The transformation engine checks each node in the plan tree to see which query transformations apply to that node. The query transformations implement the logic that transforms an IBM Cognos query into one or more SQL or MDX queries that the target data source(s) can understand. The transformations also add nodes representing any data manipulation and local processing operations that might be required to produce the requested result.

The transformations occur in several passes and potentially several iterations per pass until all possible transformations have been applied. During this process, the transformation engine connects to the IBM Cognos 10 Content Manager to look up model information that applies to the
query being processed. When all transformations have been applied, the plan tree has morphed into a run tree, and is ready for execution.

The run tree is at the heart of query execution. Results flow from the leaf nodes of the run tree to the root node where the result is represented in a format suitable for the report service to perform rendering of the report output. A run tree consists of various types of nodes, each type representing a different function:

- SQL execution
- MDX execution
- data manipulation
- local processing

In the simplest form of a dimensional style query, MDX execution nodes cause the MDX engine to pull data from the Data Cache, if available. Otherwise, it sends an MDX query to an OLAP data source. The results are stored in the Data Cache and go through some data manipulation nodes in the run tree, which might alter the shape of the results. Then local processing nodes flatten the multidimensional result and sort the data before returning the requested results to the Report Service.

In a more complex query, such as a query against a DMR package, the report request is dimensional in nature, but the data source is relational. This means the query generated for the report is MDX, but the data source only understands SQL. Thus the run tree consists of a mixture of all four types of execution nodes. The execution engine first sends SQL queries to the relational data source. Local processing nodes then reshape the results into dimensional form for storage in the data cache, from which MDX nodes query data just as they would from a dimensional data provider. Subsequent execution proceeds as for a dimensional query against an OLAP data source.

Configuring Data Source Connectivity

The IBM Cognos 10 Dynamic Query Mode can utilize the following OLAP data sources as reporting databases:

- Oracle Essbase
- SAP BW
- TM1
- Microsoft SQL Server Analysis Services

For these OLAP sources, IBM Cognos 10 uses the same data source connectivity install procedure for both the Dynamic Query Mode and the Compatible Query Mode.

The IBM Cognos 10 Dynamic Query Mode can utilize the following relational data sources as reporting databases:

- IBM DB2
- Netezza
For the relational data sources, the Dynamic Query Mode requires only that you copy a Type 4 JDBC driver into the appropriate library. The Compatible Query Mode, however, requires a different connectivity install procedure, which is not covered in this document.


The following sections describe how to configure connectivity to each data source type in more detail.

**Oracle Essbase**

**Understanding How IBM Cognos 10 Connects to Oracle Essbase**

Both IBM Cognos 10 Compatible Query Mode and Dynamic Query Mode use the same Oracle Essbase client install. The IBM Cognos 10 Compatible Query Mode uses the grid API from the Oracle Essbase bin directory whereas the IBM Cognos 10 Dynamic Query Mode uses JAR files located in the Oracle Essbase JavaAPI lib directory. Both types of files are located using the Oracle Essbase environment variables created by the Oracle Essbase client install.

The table below specifies the file names and environment variables used by each of the IBM Cognos 10 query modes.

**Table 1 Environment variable and connectivity file requirements for Oracle Essbase connectivity**

<table>
<thead>
<tr>
<th>Query mode</th>
<th>Oracle Essbase 9.3.X</th>
<th>Oracle Essbase 11.1.X</th>
</tr>
</thead>
<tbody>
<tr>
<td>IBM Cognos 10 Compatible Query Mode</td>
<td>ARBORPATH Essapinu*</td>
<td>ESSBASEPATH Essapinu*</td>
</tr>
<tr>
<td>IBM Cognos 10 Dynamic Query Mode</td>
<td>ARBORPATH Ess_es_server.jar Ess_japi.jar</td>
<td>ARBORPATH Essapinu* Ess_japi.jar Cpld14.jar</td>
</tr>
</tbody>
</table>

When IBM Cognos 10 connects to an Oracle Essbase 9.3.X data source, it will use the ARBORPATH for both query modes to locate the client libraries. However, when using IBM Cognos 10 against an Oracle 11.1.X data source, Compatible Query Mode queries will use the ESSBASEPATH, while Dynamic Query Mode queries will use the ARBORPATH. Typically the ESSBASEPATH and ARBORPATH will be set to the same location within the Oracle Essbase install.

**Configuring Connectivity to Oracle Essbase**
To configure Oracle Essbase connectivity for use with IBM Cognos 10 installed on Microsoft Windows, follow the step-by-step instructions below. They assume that the Oracle Essbase client was successfully installed.

1. From the Start\Run menu type in cmd and hit the Enter key. This will bring up a command prompt window.
2. Within the command prompt window, type in esscmd and press the Enter key. If the Oracle Essbase client was installed successfully, the Oracle Essbase command prompt should launch and display the version. The image below illustrates the esscmd command executed in a DOS window. The command responds with an esscmd command prompt displaying the Oracle Essbase version. In this case the version is 9.3.1.

Figure 2 esscmd Command Window Displaying the Oracle Essbase Version

3. If the Oracle Essbase release version is 11.1.2, no further configuration is required. If the release version is 9.3.X or 11.1.1 then proceed to the next steps.
4. Locate the c10\configuration\qfs_config.xml file and make a backup copy.
5. Open the original qfs_config.xml file using a text editor.
6. Locate the following section:

```xml
<provider name="DB2OlapODP" libraryName="essodp93" connectionCode="DO"/>
<provider name="DB2OlapODP" libraryName="essodp111" connectionCode="DO"/>
<provider name="DB2OlapODP" libraryName="essodp112" connectionCode="DO"/>
```
7. Remove the comment tags from the essodp93 or essodp111 provider depending on your Oracle Essbase version. For this example the Oracle Essbase version being used is 9.3.X.
8. Comment out the essodp112 provider. Once completed the entry should now read as follows:

```xml
<provider name="DB2OlapODP" libraryName="essodp93" connectionCode="DO"/>
<provider name="DB2OlapODP" libraryName="essodp111" connectionCode="DO"/>
<!--provider name="DB2OlapODP" libraryName="essodp112" connectionCode="DO"-->
```
9. Save the changes and close the file.
10. The changes to this file will be picked up once a Stop and Start is done on the IBM Cognos 10 service.

Data Source Specific Configuration Settings for Oracle Essbase

The following IBM Cognos 10 configuration settings within the eb.properties file are available when using Oracle Essbase as a data source.
Treat Nulls as Zeros within Calculations

Impacts: The result of calculations on data items that contain null data values.

Usage: This set of parameters controls whether or not null data values are treated as zeros when used in calculations. If the parameters are enabled, 100 + null would result in 100. If the parameters are disabled, 100 + null would result in null.

By default, these parameters are disabled.

Interoperability with other parameters: None

Setting these parameters: The parameters are available within the c10/configuration/xqe/eb.properties file as shown here (with the default settings):

null.plus.operator=null
null.minus.operator=null
null.multiply.operator=null
null.divide.numerator=null
null.divide.denominator=null
null.modulo.dividend=null
null.modulo.divisor=null

To enable this feature, change the null values to zero as follows:

null.plus.operator=zero
null.minus.operator=zero
null.multiply.operator=zero
null.divide.numerator=zero
null.divide.denominator=zero
null.modulo.dividend=zero
null.modulo.divisor=zero

These changes will be picked up once the IBM Cognos 10 service is restarted. After the restart, this change will affect all queries against any Essbase data source through IBM Cognos 10. In a distributed environment, this change will need to be made on all IBM Cognos 10 servers performing data access.

SAP BW
Understanding How IBM Cognos 10 Connects to SAP BW

Since both IBM Cognos 10 query modes use the same SAP BW client and the same librfc32 client library, no additional configuration is required beyond the actual install of the SAP BW client. The only exception to this is covered by the following section.

Configuring Connectivity to SAP BW (64-bit only)

When IBM Cognos 10 is installed as a 64-bit application, Compatible Query Mode queries will require the 32-bit librfc32 client library and Dynamic Query Mode queries will require the 64-bit librfc32 client library. Since both 32 and 64-bit libraries have the same name, the only way to tell them apart is by their file size. The 64-bit library will have the larger file size.
To enable SAP BW connectivity for both Compatible Query Mode and Dynamic Query Mode queries when IBM Cognos 10 is installed as a 64-bit application, follow these steps:

1. Obtain both the 32-bit and 64-bit librfc client libraries from the SAP BW Administrator or SAP Marketplace. The 64-bit librfc library must have a version of 7.10 or lower.
2. If the library is compressed using SAPCAR, use the following command to decompress it:
   
   ```bash
   sapcar -xvf librfxxxxxx.sar
   ```
3. Copy the 32-bit library into the c10\bin directory.
4. Copy the 64-bit library into the c10\bin64 directory.
5. The changes to this file will be picked up once a Stop and Start is done on the IBM Cognos 10 service.

**Data Source Specific Configuration Settings for SAP BW**

The following IBM Cognos 10 configuration settings within the bw.properties file are available when using SAP BW as a data source.

**Treat Nulls as Zeros within Calculations**

**Impacts:** The result of calculations on data items that contain null data values.

**Usage:** This set of parameters control whether or not null data values are treated as zeros when used in calculations. If the parameters are enabled, 100 + null would result in 100. If the parameters are disabled, 100 + null would result in null.

By default, these parameters are disabled.

**Interoperability with other parameters:** None

**Setting these parameters:** The parameters are available within the c10/configuration/xqe/bw.properties file as shown here (with the default settings):

```properties
null.plus.operator=null
null.minus.operator=null
null.multiply.operator=null
null.divide.numerator=null
null.divide.denominator=null
null.modulo.dividend=null
null.modulo.divisor=null
```

To enable this feature, change the null values to zero as follows:

```properties
null.plus.operator=zero
null.minus.operator=zero
null.multiply.operator=zero
null.divide.numerator=zero
null.divide.denominator=zero
null.modulo.dividend=zero
null.modulo.divisor=zero
```
These changes will be picked up once the IBM Cognos 10 service is restarted. After the restart, this change will affect all queries against any SAP BW data source through IBM Cognos 10. In a distributed environment, this change will need to be made on all IBM Cognos 10 servers performing data access.

**IBM Cognos TM1**

**Understanding How IBM Cognos 10 Connects to IBM Cognos TM1**

For this data source, only the IBM Cognos 10 installs for Windows require the installation of the IBM Cognos TM1 client. The IBM Cognos 10 UNIX installs contain the IBM Cognos TM1 client software as part of the install package. This means no additional configuration or installs are required and IBM Cognos 10 should be able to connect to IBM Cognos TM1 out of the box. On Windows, IBM Cognos 10 uses a registry setting created by performing a custom install of only the IBM Cognos TM1 client from the IBM Cognos TM1 server install media to locate the TM1API.dll. This dll in turn enables IBM Cognos 10 to connect to the cubes on an IBM Cognos TM1 server.

**Configuring Connectivity to IBM Cognos TM1**

To enable IBM Cognos TM1 connectivity for both Compatible Query Mode and Dynamic Query Mode queries when IBM Cognos 10 is installed on a Windows operating system, follow these step-by-step instructions:

1. After downloading the IBM Cognos TM1 9.5.1 Server install package, extract the contents of the archive to a directory.
2. Within the directory created in the previous step, double click on setup.exe to initiate the installation procedure.
3. Once the upgrade warning message has been thoroughly read, press the OK button to continue.
4. Click Next.
5. If the license agreement is acceptable, select the I accept… radio button and then click the Next button to continue with the install.
6. From the available product selection, ensure the TM1 product is selected before clicking the Next button. The following image displays the IBM Cognos TM1 installation wizard with the radio button selected for TM1, indicating to the installer that only the TM1 server and clients should be installed.
7. Thoroughly read the .NET Framework warning button before clicking the OK button.
8. Choose an install path outside of the IBM Cognos 10 directory structure. For this example the Install to directory will be C:\Program Files\Cognos\TM1. From the available menu option, select the Custom Installation type and click the Next button to proceed. The following image displays the IBM Cognos TM1 installation wizard with the Custom installation type radio button selected. This selection indicates to the installer that the user is going to select only specific components of the install.

**Figure 4 IBM Cognos TM1 Install Screen Displaying the Install Path and the Custom Install Selection**

9. From the available install components, ensure that all the components except the TM1 OLEDB Provider are omitted from the install. Click the Next button to proceed. The following image displays the IBM Cognos TM1 installation wizard custom component selection. All elements except the TM1 OLEDB Provider are de-selected. This indicates to the installer that only the IBM Cognos TM1 OLEDB provider is to be installed.
10. The following image displays the IBM Cognos TM1 installation wizard TM1 Client Configuration selection. The Admin Server Host machine name has been cleared and the Disable Excel edit in cell capability, the set TM1 to Autoload in Excel and the Use Integrated Login options have all been unchecked. Click the Next button.

**Figure 5 IBM Cognos TM1 Install Screen With Only the TM1 OLEDB Provider Selected for Install**

11. Click the Install button to finish the install.

12. This client library will be picked up once a Stop and Start is done on the IBM Cognos 10 service.

**Data Source Specific Configuration Settings for IBM Cognos TM1**

The following IBM Cognos 10 configuration settings within the qfs_config.xml file are available when using IBM Cognos TM1 as a data source.

*UseNonEmptyOnDataQueryThreshold*
As of IBM Cognos Business Intelligence 10.1.1, this parameter has been deprecated in the Dynamic Query Mode. Overriding the default value is no longer necessary.

**ConvertNullCellsToUndefVals**

As of IBM Cognos Business Intelligence 10.1.1, this parameter has been deprecated in the Dynamic Query Mode. Overriding the default value should be avoided.

**UseProviderCrossJoinThreshold**

**Impacts:** May improve performance of reports against sparsely populated TM1 cubes.

**Usage:** This parameter controls whether combinations of members on an edge, which have no measure values, are retrieved from the TM1 server.

By default, this parameter is set to 0, which disables the feature. A value greater than one enables the feature. Combination sets whose size is below the threshold are not affected.

**Interoperability with other parameters:** Enable this feature only with the default setting of the ConvertNullCellsToUndefVals parameter.

**Setting this parameter:** This parameter is available within the c10/configuration/qfs_config.xml file under the TM1OlapODPXQE provider.

```
<parameter name="UseProviderCrossJoinThreshold" value="1000"/>
```

Changes to this file take effect once IBM Cognos 10 has been restarted and apply to all queries in the Dynamic Query Mode for each IBM Cognos 10 installation. In a distributed environment with multiple installations of IBM Cognos BI sharing a common content store, make sure that all have the same setting.

Guidance: Setting the threshold too large will cause few reports to see the benefit. Setting it too small will cause some reports to perform worse, especially against data that is not sparse. The ideal value for this setting will vary depending on the environment; a good starting point would be 1000.

This feature is in effect only for reports that have null suppression enabled on both rows and columns. When the feature is in effect, it is possible to get inconsistent results in some cases. Because the performance of null suppression has been improved, reports that previously required this feature may now have acceptable performance without it. Enabling this feature is recommended only if all reports return the same results with and without enabling the feature.

**Microsoft SQL Server Analysis Services**

**Understanding How IBM Cognos 10 Connects to Microsoft SQL Server Analysis Services**

Both IBM Cognos 10 Compatible Query Mode and Dynamic Query Mode use the OLE DB Provider client software for Microsoft SQL Server Analysis Services to connect to Analysis Services.
Services data sources. There are numerous versions of this client software available from Microsoft. Select the version that matches the version of the Analysis Services data source of interest in relevant updates, patches, revisions, and service packs.

In addition, observe the following guidelines:

- For IBM Cognos 10 Compatible Query Mode, the 32-bit Microsoft SQL Server Analysis Services client should be always used, as the 64-bit client is not supported by this query mode.
- For IBM Cognos 10 Dynamic Query Mode, the 64-bit client should be used if the IBM Cognos 10 installation is 64-bit, otherwise the 32-bit client should be used.
- If both Dynamic Query Mode and Compatible Query Mode will be used on the same machine, it is recommended that a 32-bit installation (for both the Microsoft SQL Server Analysis Services client software and for IBM Cognos 10 itself) be used, as Compatible Query Mode only supports the 32-bit client software, and it is not recommended to install both the 32-bit and 64-bit client software for the same version of Analysis Services on the same machine.
- Both IBM Cognos 10 Compatible Query Mode and Dynamic Query Mode support connectivity to Microsoft SQL Server Analysis Services versions 2005 (SP3 or higher recommended), 2008 and 2008 R2 (recommended). However, only Compatible Query Mode supports Analysis Services 2000 (SP4 or higher recommended).
- Depending on the current software environment on the machine, other Microsoft pre-requisite software such as Microsoft Core XML Services may be required for the client installation to succeed. Refer to Microsoft product documentation as needed for instructions.

### Configuring Connectivity to Microsoft SQL Server Analysis Services

Once the required Microsoft Analysis Services OLE DB Provider client software has been installed, the IBM Cognos 10 Service must be stopped and restarted.

### Data Source Specific Configuration Settings for Microsoft SQL Server Analysis Services

IBM Cognos 10 supports several Microsoft SQL Server Analysis Services provider-specific configuration settings, which are used to set properties in the data source connection string. For the Dynamic Query Mode, these configuration settings belong in c10/configuration/xqeodp.config.xml. If this file does not exist, create it by copying the sample file c10/configuration/xqeodp.config.xml-example and modify settings as required. For IBM Cognos 10 Compatible Query Mode, these configuration settings can be found in c10/configuration/qfs.config.xml. For both query modes, there are separate settings for each version of Analysis Services. For full details on how these properties affect the behaviour of Microsoft SQL Server Analysis Services, refer to the appropriate Microsoft documentation.

**Connection Timeout**

This value must be between 30 and 180 seconds, inclusive.

**Generic Timeout**
Value is in seconds, and must be at least 0.

**Command Timeout**

Value is in seconds. Default is -1, which means it will use the default value from the Microsoft SQL Server Analysis Services configuration. 0 means no timeout.

**Maximum String Length**

Maximum length of Microsoft SQL Server Analysis Services string data in bytes. Default and minimum values are 1024.

**Application Name**

Name of the application to be displayed when Microsoft SQL Server Tracing is used. This setting is only available for Dynamic Query Mode. The value is pre-set to a hardcoded string in Compatible Query Mode.

**IBM DB2**

**Understanding How IBM Cognos 10 Connects to IBM DB2**

For this data source, the IBM Cognos 10 Compatible Query Mode requires the installation of the IBM DB2 client software. The IBM Cognos 10 Dynamic Query Mode on the other hand only requires access to the IBM DB2 type 4 JDBC driver and its appropriate license file. The following table lists the type of IBM DB2 databases and the required license file name.

<table>
<thead>
<tr>
<th>IBM DB2 Database</th>
<th>License File Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>DB2 UDB for Linux, UNIX and Windows</td>
<td>db2jcc_license_cu.jar</td>
</tr>
<tr>
<td>DB2 UDB for Linux, UNIX, Window, z/OS</td>
<td>db2jcc_license_cisuz.jar</td>
</tr>
</tbody>
</table>

**Configuring Connectivity to IBM DB2**

To configure IBM DB2 connectivity for use within IBM Cognos 10 installed on Microsoft Windows, follow these instructions:

1. Within the IBM DB2 install directory of the database to be used for the connection, locate the ..\SQLIB\JAVA directory.
2. Within this directory locate and copy the db2jcc4.jar and the db2jcc_license_cu.jar files.
3. Within the IBM Cognos 10 install directory, locate the ..\v5datasever\lib and the ..\p2pd\web-inf\lib directory.
4. Paste the db2jcc4.jar and db2jcc_license_cu.jar files into both these directories.
5. In order for the IBM DB2 driver to be picked up by IBM Cognos 10, the IBM Cognos 10 service will need to be stopped and started.
Netezza

Understanding How IBM Cognos 10 Connects to Netezza

For this data source the IBM Cognos 10 Compatible Query Mode uses the Netezza ODBC driver to connect while the IBM Cognos 10 Dynamic Query Mode uses the type 4 JDBC driver.

Configuring Connectivity to IBM Netezza

To configure Netezza connectivity for use within IBM Cognos 10 installed on Microsoft Windows, follow these instructions:

1. Within the Netezza client install directory, locate and copy the nzjdbc.jar.
2. Within the IBM Cognos 10 install directory, locate the ..\v5dataserver\lib and the ..\p2pd\web-inf\lib directory.
3. Copy the nzjdbc.jar file into both these directories.
4. In order for the Netezza driver to be picked up by IBM Cognos 10, the IBM Cognos 10 service will need to be stopped and started.

Microsoft SQL Server

Understanding How IBM Cognos 10 Connects to Microsoft SQL Server

IBM Cognos 10 Compatible Query Mode can connect to Microsoft SQL Server through Microsoft SQL Server ODBC, OLE-DB or via the Microsoft SQL Server native client. For any of these connection types, the IBM Cognos 10 Compatible Query Mode requires that the client is installed on the same computer as the IBM Cognos 10 software. The file requirements for the IBM Cognos Dynamic Query Mode are dependant on the data source security strategy. For non-integrated security connections that pass the saved signon information, the IBM Cognos 10 Dynamic Query Mode only requires access to the Microsoft Type 4 JDBC driver. For integrated security connections that use the service credentials to connect to the data source, IBM Cognos Dynamic Query Mode requires access to both the Microsoft Type 4 JDBC driver and its associated 32 or 64 bit authentication dynamic linked library (DLL). The following table lists the type of IBM Cognos BI data source authentication types and the files required to establish a successful connection.

Table 3 Files required to connect to Microsoft SQL Server via the various IBM Cognos 10 authentication types

<table>
<thead>
<tr>
<th>IBM Cognos 10 Authentication Type</th>
<th>Connectivity File(s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>No authentication</td>
<td>sqljdbc4.jar</td>
</tr>
<tr>
<td>IBM Cognos software service credentials</td>
<td>sqljdbc4.jar</td>
</tr>
<tr>
<td></td>
<td>sqljdbc_auth.dll</td>
</tr>
<tr>
<td>An external namespace</td>
<td>sqljdbc4.jar</td>
</tr>
<tr>
<td></td>
<td>sqljdbc_auth.dll</td>
</tr>
<tr>
<td>The signons of this connection</td>
<td>sqljdbc4.jar</td>
</tr>
</tbody>
</table>
Configuring Connectivity to Microsoft SQL Server

To configure Microsoft SQL Server Connectivity for the “IBM Cognos software service credentials” or “An external namespace” authentication type, follow these instructions:

2. Within the Microsoft SQL Server JDBC driver install directory, locate and copy the sqljdbc4.jar file.
3. Within the IBM Cognos 10 install directory, locate the ..\v5dataserver\lib and the ..\p2pd\web-inflib directory.
4. Copy the sqljdbc4.jar file into both these directories.
5. Locate the c10\v5dataserver\databaseDriverLocations.properties.sample file and rename it to databaseDriverLocations.properties.
6. Open the newly renamed databaseDriverLocations.properties file using a text editor.
7. Set the databaseJNIPath to the location of the sqljdbc_auth.dll file. For this example the databaseJNIPath will be D://MSSQL_AUTH. The completed entry would represent the following text:
   
   databaseJNIPath=D:\MSSQL_AUTH

8. Save the changes and close the file.
9. In order for the Microsoft SQL Server driver to be picked up by IBM Cognos 10, the IBM Cognos 10 service will need to be stopped and started.

To configure Microsoft SQL Server Connectivity for “Signon for this connection” or “No authentication” authentication type, follow these instructions:

2. Within the Microsoft SQL Server JDBC driver install directory, locate and copy the sqljdbc4.jar file.
3. Within the IBM Cognos 10 install directory, locate the ..\v5dataserver\lib and the ..\p2pd\web-inflib directory.
4. Copy the sqljdbc4.jar file into both these directories.
5. In order for the Microsoft SQL Server driver to be picked up by IBM Cognos 10, the IBM Cognos 10 service will need to be stopped and started.

NCR Teradata

Understanding How IBM Cognos 10 Connects to NCR Teradata

For this data source the IBM Cognos 10 Compatible Query Mode uses the NCR Teradata ODBC driver to connect while the IBM Cognos 10 Dynamic Query Mode uses the type 4 JDBC driver and its required configuration file.

Configuring Connectivity to NCR Teradata
To configure NCR Teradata connectivity for use within IBM Cognos 10 installed on Microsoft Windows, follow these instructions:

1. Within the NCR Teradata install directory of the database to be used for the connection, locate and copy the terajdbc4.jar and tdgssconfig.jar files.
2. Within the IBM Cognos 10 install directory, locate the ..\v5dataserver\lib and the ..\p2pd\web-inf\lib directory.
3. Paste the terajdbc4.jar and tdgssconfig.jar files into both these directories.
4. In order for the NCR Teradata driver to be picked up by the IBM Cognos 10 The IBM Cognos 10 service will need to be stopped and started.

Oracle

Understanding How IBM Cognos 10 Connects to Oracle

For this data source, the IBM Cognos 10 Compatible Query Mode requires the installation of the Oracle client software. The IBM Cognos 10 Dynamic Query Mode has the ability to use the same Oracle JDBC driver to perform either type 2 or the type 4 Oracle JDBC connections. The name of the Oracle JDBC driver depends on the version of Java being used within the IBM Cognos 10 install. The following table provides a list of Oracle JDBC driver names in relation to the Java version.

### Table 4 Oracle JDBC driver names for the Java version being used

<table>
<thead>
<tr>
<th>Java Version</th>
<th>Oracle JDBC Driver Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.5</td>
<td>ojdbc5.jar</td>
</tr>
<tr>
<td>1.6</td>
<td>ojdbc6.jar</td>
</tr>
</tbody>
</table>

When using the Dynamic Query Mode to perform a type 4 Oracle JDBC connection, there is no requirement to have the Oracle native libraries installed. However, when using the Dynamic Query Mode to perform a type 2 Oracle JDBC connection, the Oracle native libraries need to be installed. Depending on the operating system, the PATH, LIBPATH and LD_LIBRARY_PATH will need to be configured to the location of the ociJDBCXX (where the XX represents the Oracle version) library. The Dynamic Query Mode also requires that the ..\c10\v5dataserver\databaseDriverLocations.properties.sample file is renamed to databaseDriverLocations.properties and the databaseJNIPath is set to the same location as the above mentioned environment variable.

Configuring Connectivity to Oracle

To configure Oracle connectivity for use within IBM Cognos 10 installed on Microsoft Windows using a 1.6 version of Java, follow these instructions:

1. Within the Oracle install directory of the database to be used for the connection, locate and copy the ojdbc6.jar file.
2. Within the IBM Cognos 10 install directory, locate the ..\v5dataserver\lib and the ..\p2pd\web-inf\lib directory.
3. Paste the ojdbc6.jar file into both these directories.
4. In order for the Oracle driver to be picked up by IBM Cognos 10, the IBM Cognos 10 service will need to be stopped and started.

**Publishing Packages for Dynamic Query Mode**

As stated in Section 4, the IBM Cognos 10 Dynamic Query Mode can utilize the following OLAP data sources as reporting databases:

1. Oracle Essbase
2. SAP BW
3. IBM Cognos TM1
4. Microsoft SQL Server Analysis Services

The IBM Cognos 10 Dynamic Query Mode can utilize the following relational data sources as reporting databases:

1. IBM DB2
2. Netezza
3. Microsoft SQL Server
4. NCR Teradata
5. Oracle

If you create a dimensionally modeled relational (DMR) package in IBM Cognos Framework Manager, the Dynamic Query Mode uses the dimensional layer of the model to provide OLAP query behavior over these relational data sources.

In order to create a reporting application (consisting of one or more packages and their reports) for the Dynamic Query Mode, you must create a new model in Framework Manager. Refer to the [IBM Cognos Framework Manager User Guide](http://ibm.com/support/docview.wss?uid=swg27018249) for guidelines on relational and dimensional modeling.

We recommend that you enable the Dynamic Query Mode when creating a new project in Framework Manager, provided that your model will contain only data sources that are supported by the Dynamic Query Mode. On the **New Project** window, select the **Use Dynamic Query Mode** checkbox. This ensures that all modeling and publishing activities use the Dynamic Query Mode. Furthermore, all reports based on packages published from the project run in the Dynamic Query Mode.

If your modeling project uses the Compatible Query Mode, you still have the option to test queries and publish packages using the Dynamic Query Mode. However, the option to test using the Dynamic Query Mode (DQM) will only be available for supported data sources. When publishing a supported DQM package from a project that uses the Compatible Query Mode, modelers can decide which query mode will be used when running reports against that package. If you wish to switch the query mode for a package, you must republish the package with the desired setting.

Packages must contain only supported Dynamic Query Mode data sources to enable Dynamic Query Mode. If unsupported data sources are included in the package, the Dynamic Query Mode option will not be available when publishing the package. In order to take advantage of DQM,
ensure that your IBM Cognos BI environment is configured for connectivity to the supported data sources. See Section 4 for more details.

The following topics in this section describe how to configure each supported DQM data source and how to publish packages that use the new query service.

You can create data source connections in either IBM Cognos Connection or through IBM Cognos BI Framework Manager. These examples show how to create them in IBM Cognos BI Framework Manager. However, the configuration steps are the same in both environments.

**Create a Project, Connection and Package for Oracle Essbase**

Ensure that the Oracle Essbase client is installed and configured for connectivity to Oracle Essbase on the IBM Cognos BI servers. If your modeling project requires the Compatible Query Mode, then the IBM Cognos BI Framework Manager machine must also have the Oracle Essbase client installed and configured.

Oracle Essbase packages must be published to IBM Cognos 10 through Framework Manager.

1. Open IBM Cognos BI Framework Manager, and then click **Create a new project**.
2. In the Project name box, type the desired name. In this case, **Oracle Essbase - GO Sales** will be used.
3. Select **Use Dynamic Query Mode** if you want all modeling activities and execution of reports based on packages published from the project to use the Dynamic Query Mode, as recommended.
4. Click **OK**. The Select Languages dialog box appears.
5. Choose the desired design language and click OK. In this example, **English** is selected. The Metadata Wizard appears.
6. Ensure **Data Sources** is selected, and then click **Next**.
7. Click the **New** button to create a new data source connection.
8. In the **New Data Source** wizard, click **Next**, in the **Name** box, type **Essbase**, and then click **Next**.

   **Figure 7 New Data Source Wizard with Oracle Essbase selected**

   - **Type**: Oracle Essbase
   - **Isolation Level**: User the default object gateway
   - **Cursor Stability**:

9. Under **Type**, select **Oracle Essbase**.
10. Click **Next**.
11. Based on the connection information provided by the Oracle Essbase administrator, type in the **Server name** and configure the **Signon** credentials.

**Figure 8 New Data Source Wizard with signon information provided**

12. Click **Test the connection**, and then click **Test**.

On the results page of the connection test, notice the results showing a status of Succeeded for both Compatible and Dynamic query modes.

**Figure 9 Test the connection screen showing successful Compatible and Dynamic query mode test results**

13. Click **Close**, click **Close** again, and then click **Finish**.
14. Click **Close**.

The new data source appears in the list as seen below.

**Figure 10 Metadata Wizard showing new Essbase data source**

The next step will be to import a cube and publish it to IBM Cognos 10.

15. Ensure the Oracle Essbase data source that was created is selected, click **Next**, and then locate and select the desired cube.
16. Click Next, and then select the desired language for the cube and how attribute dimensions should be presented. They can be presented as either separate dimensions or properties of the dimension they are associated with.

**Figure 11 Metadata Wizard showing cube options for language and presenting attribute dimensions**

17. Click Next, leave the **Create a default package** option selected, and then click Finish.

18. In the Name box, type an appropriate name for the package. In this case the name **Oracle Essbase - GO Sales** will be used. Click Finish, and then click Yes to open the Publish Wizard.

19. Follow the wizard instructions making the appropriate configurations required and click **Next** until the Options screen is reached. Notice the **Use Dynamic Query Mode** option.

**Figure 12 Publish Wizard showing the Use Dynamic Query Mode option**

20. This option is available when both of the following conditions apply:
   - The Query Mode property of the project is set to **Compatible**
   - The package contains only supported data sources
21. Select the **Use Dynamic Query Mode** option, if available.
22. Click **Publish**, and then click **Finish**.

The package is now available in IBM Cognos 10 and will use the Dynamic Query Mode for reports. In IBM Cognos Connection, the type of query mode used by the package can be verified in the package properties.

**Figure 13 Package properties showing a Query Mode of Dynamic**

Create a Project, Connection and Package for SAP BW

Ensure that the SAP GUI is installed and configured for connectivity to SAP BW on the IBM Cognos BI servers. If your modeling project requires the Compatible Query Mode, then the IBM Cognos BI Framework Manager machine must also have the SAP GUI installed and configured.

An SAP BW package can be published directly from IBM Cognos Connection or through Framework Manager. However, importing SAP BW metadata into Framework Manager allows for additional modeling and testing before the package is published, as shown in this example. For information on publishing SAP BW packages directly in IBM Cognos Connection, please see the IBM Cognos BI Administration and Security Guide.

1. Open IBM Cognos BI Framework Manager, and then click Create a new project.
2. In the **Project name** box, type the desired name. In this case, SAP BW - GO Sales will be used.
3. Select **Use Dynamic Query Mode** if you want all modeling activities and execution of reports based on packages published from the project to use the Dynamic Query Mode, as recommended.
4. Click **OK**.
   The Select Languages dialog box appears.
5. Choose the desired design language and click OK. In this example, **English** is selected.
   The Metadata Wizard appears.
6. Ensure **Data Sources** is selected, and then click **Next**.
7. Click the **New** button to create a new data source connection.
8. In the **New Data Source** wizard, click **Next**, in the **Name** box, type **SAP BW**, and then click **Next**.
9. Under **Type**, select **SAP BW**.

   **Figure 14 New Data Source Wizard with SAP BW selected**

10. Click **Next**.
11. Based on the connection information provided by the SAP BW administrator, select the **SAP logon type**, type in the **Application server name**, **System number**, **Client number** and provide the security signon configuration.
Figure 15 New Data Source Wizard with SAP BW connection information provided

12. Click **Test the connection**, and then click **Test**.
On the results page of the connection test, notice the results showing a status of Succeeded for both Compatible and Dynamic query modes.

Figure 16 Test the connection screen showing successful Compatible and Dynamic query mode test results

13. Click **Close**, click **Close** again, and then click **Finish**.
14. Click **Close**.
The new data source appears in the list.
The next step will be to import SAP BW metadata.

15. Ensure the SAP BW data source that was created is selected, click **Next**, and then locate and select the desired reporting objects (InfoQuery, InfoCube, etc) for import.

16. Click **Next**, add the desired languages, and then click **Next**.

17. On the **Generate Dimensions** screen, select how you want to display object names and organize the dimensions.
   
   You have the choice to display object names as short name, long name or technical name and you can choose to enhance the model for SAP BW organization of objects.

18. Click **Next** to import the metadata, and then click **Finish**.

19. In the **Project Viewer**, expand the new namespace created for the SAP BW metadata and notice all the dimensions and key figures have been imported.
If the **Query Mode** property of the project is set to **Dynamic** when testing these SAP BW objects, the test queries will run in Dynamic Query Mode. Otherwise, there is an option to use the Dynamic Query Mode instead of Compatible query mode on the Test tab in the lower left corner.

For more information about working with SAP BW metadata, please refer to the IBM Cognos BI Framework Manager User Guide.

In the next steps, a package will be created and published.

20. In the **Project Viewer**, right-click **Packages**, point to **Create**, and then click **Package**.
21. In the **Name** box, type an appropriate name for the package. In this case SAP BW - GO Sales will be used. Click **Next**.
22. Select the objects to include in the package.

![Figure 21 Create Package dialog showing selected model objects](image)

23. Click **Next**, click **Finish**, and then click **Yes** to open the **Publish Wizard**.
24. Follow the wizard instructions making the appropriate configurations required and click **Next** until the **Options** screen is reached.
   Notice the **Use Dynamic Query Mode** checkbox.
25. Select the **Use Dynamic Query Mode** option, if available.
26. Click **Publish**, and then click **Finish**.
   The package is now available in IBM Cognos 10 and will use the Dynamic Query Mode for reports and analyses.

### Create a Project, Connection and Package for IBM Cognos TM1

Ensure that the IBM Cognos TM1 client is installed and configured for connectivity to TM1 on the IBM Cognos BI servers. If your modeling project requires the Compatible Query Mode, then the IBM Cognos BI Framework Manager machine must also have the TM1 client installed and configured.

IBM Cognos TM1 packages must be published through IBM Cognos BI Framework Manager.

1. Open IBM Cognos BI Framework Manager, and then click **Create a new project**.
2. In the Project name box, type the desired name. In this case, **IBM Cognos TM1 - GO Sales** will be used.
3. Select **Use Dynamic Query Mode** if you want all modeling activities and execution of reports based on packages published from the project to use the Dynamic Query Mode, as recommended.
4. Click **OK**.
   The Select Languages dialog box appears.
5. Choose the desired design language and click **OK**. In this example, **English** is selected.
   The Metadata Wizard appears.
6. Ensure **Data Sources** is selected, and then click **Next**.
7. Click the **New** button to create a new data source connection.
8. In the **New Data Source** wizard, click **Next**, in the **Name** box, type **IBM Cognos TM1**, and then click **Next**.
9. Under **Type**, select **IBM Cognos TM1**.
10. Click **Next**.

11. Based on the connection information provided by the IBM Cognos TM1 administrator, type in the **Administration Host**, **Server Name**, and **Signon** credentials. The Administration Host is the name of the physical machine hosting the IBM Cognos TM1 server(s). The Server Name refers to the name of the cube being served by an IBM Cognos TM1 server on the administration host machine.

**Figure 24 New Data Source Wizard with connection information provided**

12. Click **Test the connection**, and then click **Test**.

On the results page of the connection test, notice the results showing a status of Succeeded for both Compatible and Dynamic Query Modes.
13. Click **Close**, click **Close** again, and then click **Finish**.

14. Click **Close**.

The new data source appears in the list.

**Figure 26 Metadata Wizard showing new IBM Cognos TM1 data source**

15. Ensure the IBM Cognos TM1 data source that was created is selected, click **Next**, and then select the cube for import.

**Figure 27 Metadata Wizard showing a selected cube**

16. Click **Next**, and if required, select each dimension and the Alias tables language you wish to import.
17. Click **Next**, leave the **Create a default package** option selected, and then click **Finish**.

18. In the **Name** box, type an appropriate name for the package, in this case **IBM Cognos TM1 - GO Sales** will be used, click **Finish**, and then click **Yes** to open the **Publish Wizard**.

19. Follow the wizard instructions making the appropriate configurations required and click **Next** until the **Options** screen is reached.

   Notice the **Use Dynamic Query Mode** checkbox.

**Figure 29 Publish Wizard showing the Use Dynamic Query Mode option**

   This option is available when both of the following conditions apply:
   - The Query Mode property of the project is set to Compatible
   - The package contains only supported data sources

20. Select the **Use Dynamic Query Mode** option, if available.

21. Click **Publish**, and then click **Finish**.

   The package is now available in IBM Cognos 10 and will use the Dynamic Query Mode for reports and analyses.
Create a Project, Connection and Package for Microsoft SQL Server Analysis Services

Ensure that the appropriate OLE DB Provider client software for your version of Microsoft SQL Server Analysis Services is installed and configured for connectivity to Analysis Services on the IBM Cognos BI servers. If your modeling project requires the Compatible Query Mode, then the IBM Cognos BI Framework Manager machine must also have the Analysis Services client installed and configured.

Microsoft SQL Server Analysis Services packages must be published through IBM Cognos BI Framework Manager.

1. Open IBM Cognos BI Framework Manager, and then click **Create a new project**.
2. In the **Project name** box, type the desired name. In this example, **MSAS - GO Sales** is used.
3. Select **Use Dynamic Query Mode** if you want all modeling activities and execution of reports based on packages published from the project to use the Dynamic Query Mode, as recommended.
4. Click **OK**.
   The **Select Languages** dialog box will appear.
5. Choose the desired design language and click **OK**. In this example, **English** is selected.
   The **Metadata Wizard** appears.
6. Ensure **Data Sources** is selected, and then click **Next**.
7. Click the **New** button to create a new data source connection.
8. In the **New data source wizard**, click **Next**. In the **Name** box, type the desired data source name (**MSAS** in this example) then click **Next**.
9. Under **Type**, select the version of Analysis Services to be used. In this example, **Microsoft Analysis Services 2008** is selected.
10. Click **Next**.
11. Based on the connection information provided by the Microsoft SQL Server Analysis Services administrator, type in the **Server Name** and **Instance Name**, if applicable, select the authentication mode to be used, and configure the namespace or signon credentials if applicable.
12. Click **Test the connection**, and then click **Test**.  
On the results page of the connection test, notice the results showing a status of **Succeeded** for both Compatible and Dynamic query modes.

**Figure 32 Test the connection screen showing successful Compatible and Dynamic query mode test results**

13. Click **Close**, click **Close** again, and then click **Finish**.
14. Click **Close**.  
The new MSAS data source now appears in the list in the **Select Data Source** dialog. The next steps describe how to import a cube and publish it to IBM Cognos 10.
15. Ensure that the newly created MSAS data source is selected, click Next, then locate and select the desired cube.

16. Click Next, leave the Create a default package option selected, and then click Finish.

17. In the Name box, type an appropriate name for the package (such as MSAS – GO Sales). Click Finish and then click Yes to open the Publish Wizard.

18. Follow the wizard instructions, making the appropriate configurations required and click Next until the Options screen is reached.

Notice the Use Dynamic Query Mode option.

**Figure 33 Publish Wizard showing the Use Dynamic Query Mode option**

This option is available when both of the following conditions apply:

- The Query Mode property of the project is set to Compatible
- The package contains only supported data sources

19. Select the Use Dynamic Query Mode option, if available.

20. Click Publish and then click Finish.

The package is now available in IBM Cognos 10, and will use Dynamic Query Mode for reports. In IBM Cognos Connection, the type of query mode used by the package can be verified in the package properties.
Create a Project, Connection and Package for IBM DB2

For an IBM DB2 data source, the IBM Cognos Framework Manager machine requires no additional software if all your Framework Manager projects use the Dynamic Query Mode. If any of your projects use the Compatible Query Mode, however, an IBM DB2 client must be installed and configured on the IBM Cognos BI Framework Manager machine.

The following instructions use the sample Great Outdoors Sales IBM DB2 database called GS_DB to illustrate creating a relational data source connection. This database ships with the product as part of the samples.

1. Open IBM Cognos BI Framework Manager, and then click Create a new project.
2. In the Project name box, type DB2 DQM Model.
3. Select Use Dynamic Query Mode if you want all modeling activities and execution of reports based on packages published from the project to use the Dynamic Query Mode, as recommended.
4. Click OK.
   The Select Languages dialog box appears.
5. Ensure that English is selected, and then click OK.
   The Metadata Wizard appears.
6. Ensure Data Sources is selected, and then click Next.
7. Click the **New** button to create a new data source connection.
8. In the **New Data Source** wizard, click **Next**, in the **Name** box, type `GOSALES(DB2)`, and then click **Next**.
9. Under **Type**, select **IBM DB2**.
   Notice the **Configure JDBC connection** checkbox. Ensure this box is checked so that information can be provided to connect through the JDBC driver which is required for Dynamic Query Mode.

**Figure 35 New Data Source wizard showing IBM DB2 selected and the Configure JDBC connection checkbox**

10. Click **Next**.
   In the next steps, the information provided is based on how the IBM DB2 clients on the Framework Manager machine and the IBM Cognos BI servers were configured and how security is implemented for IBM DB2. Connection information and sign on information should be provided by the database administrator.

11. In the **IBM DB2 database name** box, type `GS_DB`, and then under **Signon**, select the **Password** check box.
12. In the **User ID** box, type in the user ID, in the **Password** and **Confirm password** boxes, type in the password, and then click **Next**.
   On the next screen, the JDBC connection information will be provided.
13. In the **Server name** box, type the name of the server hosting the database, in the **Port number** box, enter the port number provided by the database administrator, and then in the **Database name** box, type `GS_DB`.

14. Click **Test the connection**, and then click **Test**. On the results page of the connection test, notice the results showing a status of Succeeded for the Dynamic Query Mode.

**Figure 38 Test the connection screen showing successful Dynamic query mode test results**

<table>
<thead>
<tr>
<th>Type/Query Mode</th>
<th>Status</th>
<th>Message</th>
</tr>
</thead>
<tbody>
<tr>
<td>IBM DB2 JDBC</td>
<td>Dynamic</td>
<td>Succeeded</td>
</tr>
</tbody>
</table>

15. Click **Close**, click **Close** again, and then click **Finish**.

16. Click **Close**.

The new data source appears in the list and is configured to query using either query mode.

**Figure 39 Metadata Wizard showing new GOSALES(DB2) data source**

The next steps will be to import metadata and test query subjects.

17. Ensure that **GOSALES(DB2)** is selected, click **Next**, expand **GOSALESDW**, and then expand **Tables**.

18. Select the following tables, and then click **Next**.

- GO_TIME_DIM
- SLS_RTL_DIM
- SLS_SALES_FACT
19. Click **Import**, and then click **Finish**.
20. In the **Project Viewer**, expand **GOSALES(DB2)**.
   The query subjects appear as child objects as shown below.

   **Figure 40 Project Viewer showing imported query subjects**

21. Double-click **GO_TIME_DIM** to open its definition, and then click the **Test** tab.
   If the **Query Mode** property of the project is set to **Dynamic** when testing a query subject, the test query will run in Dynamic Query Mode. If the **Query Mode** property is set to **Compatible**, however, there is an option to use the Dynamic Query Mode on the Test tab in the lower left corner, provided that the query subject is for a data source supported by the Dynamic Query Mode.
22. Click the **Use Dynamic Query Mode** check box, if available.
23. Click **Test Sample**.
   Framework Manager sends the test query through the IBM Cognos 10 gateway to one of the IBM Cognos BI servers, which, in turn, queries the reporting database. The data retrieved by the test query appear in the **Test results** pane.
   The image below shows the data as well as the **Use Dynamic Query Mode** checkbox in the lower left corner, which has been checked.

   **Figure 41 Query Subject Definition Test tab showing test results**

   ![Query Subject Definition Test tab showing test results](image)

   You can click on the Query Information tab to view the Cognos and Native SQL as well as the XML response from the IBM Cognos BI server.
24. Click **OK**.
You should test all your model objects against the Dynamic Query Mode to ensure that SQL generation is as expected for your requirements. If you are building a DMR model, this includes foundation objects such as Data Source and Model Query Subjects as well as Regular and Measure Dimensions.

Once you have finished building the model, you can create and publish a package that uses the Dynamic Query Mode.

25. In the **Project Viewer**, right-click **Packages**, point to **Create**, and then click **Create Package**.
26. In the **Name** box, type **GOSALES (DB2)**, click **Next**, and then click **Finish**.
   A prompt appears asking if you wish to open the **Publish Wizard**.
27. Click **Yes**, deselect **Enable model versioning**, and then click **Next** twice.
   On the **Publish Wizard - Options** screen, the **Use Dynamic Query Mode** option is available when both of the following conditions apply:
   - The Query Mode property of the project is set to Compatible
   - The package contains only supported data sources

**Figure 42 Publish Wizard showing the Use Dynamic Query Mode option**

- Select **Use Dynamic Query Mode**, if available.
- Click **Publish**, and then click **Finish**.

The package is now available in IBM Cognos 10 and will use the Dynamic Query Mode for reports written against this package. In IBM Cognos Connection, the query mode used by the package can be verified in the package properties.
For a Netezza data source, the IBM Cognos Framework Manager machine requires no additional software if all your Framework Manager projects use the Dynamic Query Mode. If any of your projects use the Compatible Query Mode, however, a Netezza client must be installed and configured on the IBM Cognos BI Framework Manager machine.

In this example a database called **GOSALES1** will be used to illustrate creating a relational data source connection.

1. Open IBM Cognos BI Framework Manager, and then click **Create a new project**.
2. In the **Project name** box, type **Netezza DQM Model**.
3. Select **Use Dynamic Query Mode** if you want all modeling activities and execution of reports based on packages published from the project to use the Dynamic Query Mode, as recommended.
4. Click **OK**.
   - The Select Languages dialog box appears.
5. Ensure that **English** is selected, and then click **OK**.
   - The Metadata Wizard appears.
6. Ensure **Data Sources** is selected, and then click **Next**.
7. Click the **New** button to create a new data source connection.
8. In the **New Data Source** wizard, click **Next**, in the **Name** box, type **GOSALES(Netezza)**, and then click **Next**.
9. Under **Type**, select **Netezza (ODBC)**.
   - Notice the **Configure JDBC connection** checkbox. Ensure this box is checked so that information can be provided to connect through the JDBC driver which is required for Dynamic Query Mode.
10. Click **Next**.

   In the next steps, the information provided is based on how the Netezza clients on the Framework Manager machine and the IBM Cognos BI servers were configured and how security is implemented for Netezza. Connection information and sign on information should be provided by the database administrator.

11. In the **ODBC Data source** box, type in the ODBC connection name you configured for Netezza, and then under **Signon**, select the **Password** check box.
12. In the **User ID** box, type in the user ID, in the **Password** and **Confirm** password boxes, type in the password, and then click **Next**.
On the next screen, the JDBC connection information will be provided.
Figure 46 New Data Source wizard showing JDBC driver parameters

13. In the **Server** name box, **Port number** box, and **Database name** box, enter the values provided by the database administrator.

14. Click **Test the connection**, and then click **Test**.

   On the results page of the connection test, notice the results showing a status of Succeeded for the Dynamic Query Mode.

   **Figure 47 Test the connection screen showing successful Dynamic query mode test results**

15. Click **Close**, click **Close** again, and then click **Finish**.

16. Click **Close**.

   The new data source appears in the list and is configured to query using either query mode.

   **Figure 48 Metadata Wizard showing new GOSALES(Netezza) data source**

17. Ensure **GOSALES(Netezza)** is selected, click **Next**, expand the database and schema, and then expand **Tables**.

18. Select the following tables, and then click **Next**.
   - ORDERDETAIL
   - ORDERHEADER
   - PRODUCT
19. Click **Import**, and then click **Finish**.
20. In the **Project Viewer**, expand **GOSALES(Netezza)**.
   The query subjects appear as child objects as shown below.

   **Figure 49 Project Viewer showing imported query subjects**

21. Double-click **PRODUCT** to open its definition, and then click the **Test** tab.
   If the **Query Mode** property of the project is set to **Dynamic** when testing a query subject, the test query will run in Dynamic Query Mode. If the **Query Mode** property is set to **Compatible**, however, there is an option to use the Dynamic Query Mode on the Test tab in the lower left corner, provided that the query subject is for a data source supported by the Dynamic Query Mode.

22. Click the **Use Dynamic Query Mode** check box, if available.
23. Click **Test Sample**.
   Framework Manager sends the test query through the IBM Cognos 10 gateway to one of the IBM Cognos BI servers, which, in turn, queries the reporting database. The data retrieved by the test query appear in the **Test results** pane.
   The image below shows the data as well as the Use Dynamic Query Mode checkbox in the lower left corner, which has been checked.
24. Click OK. You should test all your model objects against the Dynamic Query Mode to ensure that SQL generation is as expected for your requirements. If you are building a DMR model, this includes foundation objects such as Data Source and Model Query Subjects as well as Regular and Measure Dimensions.

Once you have finished building the model, you can create and publish a package that uses the Dynamic Query Mode.

25. In the Project Viewer, right-click Packages, point to Create, and then click Create Package.

26. In the Name box, type GOSALES (Netezza), click Next, and then click Finish. A prompt appears asking if you wish to open the Publish Wizard.

27. Click Yes, deselect Enable model versioning, and then click Next twice. On the Publish Wizard - Options screen, the Use Dynamic Query Mode option is available when both of the following conditions apply:
   - The Query Mode property of the project is set to Compatible
   - The package contains only supported data sources
28. Select **Use Dynamic Query Mode**, if available.
29. Click **Publish**, and then click **Finish**.

The package is now available in IBM Cognos 10 and will use the Dynamic Query Mode for reports written against this package. In IBM Cognos Connection, the query mode used by the package can be verified in the package properties.

**Figure 52 Package properties showing Query Mode as Dynamic**

---

**Create a Project, Connection and Package for Microsoft SQL Server**

For a Microsoft SQL Server data source, the IBM Cognos Framework Manager machine requires no additional software if all your Framework Manager projects use the Dynamic Query Mode. If
any of your projects use the Compatible Query Mode, however, a Microsoft SQL Server client must be installed and configured on the IBM Cognos BI Framework Manager machine.

In this example the sample Great Outdoors Sales Microsoft SQL Server database called **GOSALES_DW** will be used to illustrate creating a relational data source connection. This database ships with the product as part of the samples.

1. Open IBM Cognos BI Framework Manager, and then click **Create a new project**.
2. In the **Project name** box, type **SQL Server DQM Model**.
3. Select **Use Dynamic Query Mode** if you want all modeling activities and execution of reports based on packages published from the project to use the Dynamic Query Mode, as recommended.
4. Click **OK**.
5. The Select Languages dialog box appears.
6. Ensure that **English** is selected, and then click **OK**.
7. The Metadata Wizard appears.
8. Ensure **Data Sources** is selected, and then click **Next**.
9. Click the **New** button to create a new data source connection.
10. In the **New Data Source** wizard, click **Next**, in the **Name** box, type **GOSALES(SQL Server)**, and then click **Next**.
11. Under **Type**, select the appropriate Microsoft SQL Server connection type for your Microsoft SQL Server database. Please refer to the IBM Cognos BI Administration and Security guide for more information on Microsoft SQL Server connection types and connection parameters. In this case Microsoft SQL Server (OLE DB) will be used.
12. Notice the **Configure JDBC connection** checkbox. Ensure this box is checked so that information can be provided to connect through the JDBC driver which is required for Dynamic Query Mode.

**Figure 53 New Data Source wizard showing Microsoft SQL Server (OLE DB) selected and the Configure JDBC connection checkbox**

10. Click **Next**.

In the next steps, the information provided is based on how the Microsoft SQL Server clients on the Framework Manager machine and the IBM Cognos 10 servers were configured and how security is implemented for Microsoft SQL Server. Connection information and sign on information should be provided by the database administrator.

11. Provide the connection information for the connection type chosen, and then under **Signon**, select the **Password** check box.
Figure 54 New Data Source Wizard with server name and database name provided and the Password box selected

12. In the User ID box, type in the user ID, in the Password and Confirm password boxes, type in the password, and then click Next.
On the next screen, the JDBC connection information will be provided.
13. In the **Server name** box, **Port number** box, and **Instance name** box, enter the values provided by the database administrator. In the **Database name** box, type **GOSALES Dw** for this example.

14. Click **Test the connection**, and then click **Test**.
   On the results page of the connection test, notice the results showing a status of Succeeded for the Dynamic query mode.

**Figure 56 Test the connection screen showing successful Dynamic query mode test results**

<table>
<thead>
<tr>
<th>Type / query Mode</th>
<th>Status</th>
<th>Message</th>
</tr>
</thead>
<tbody>
<tr>
<td>Microsoft SQL Server (JDBC) / Dynamic</td>
<td>Succeeded</td>
<td></td>
</tr>
</tbody>
</table>

15. Click **Close**, click **Close** again, and then click **Finish**.

16. Click **Close**.
   The new data source appears in the list and is configured to query using either query mode.

**Figure 57 Metadata Wizard showing new GOSALES(SQL Server) data source**

The next steps will be to import metadata and test query subjects.
17. Ensure **GOSALES(SQL Server)** is selected, click **Next**, expand **gosalesdw**, and then expand **Tables**.

18. Select the following tables, and then click **Next**.
   - **GO_TIME_DIM**
   - **SLS_RTL_DIM**
   - **SLS SALES FACT**

19. Click **Import**, and then click **Finish**.

20. In the **Project Viewer**, expand **GOSALES(SQL Server)**. The query subjects appear as child objects as shown below.

   **Figure 58 Project Viewer showing imported query subjects**

![Project Viewer showing imported query subjects](image)

21. Double-click **GO_TIME_DIM** to open its definition, and then click the **Test** tab.

   If the **Query Mode** property of the project is set to **Dynamic** when testing a query subject, the test query will run in Dynamic Query Mode. If the **Query Mode** property is set to **Compatible**, however, there is an option to use the Dynamic Query Mode on the Test tab in the lower left corner, provided that the query subject is for a data source supported by the Dynamic Query Mode.

22. Click the **Use Dynamic Query Mode** check box, if available.

23. Click **Test Sample**.

   Framework Manager sends the test query through the IBM Cognos 10 gateway to one of the IBM Cognos BI servers, which, in turn, queries the reporting database. The data retrieved by the test query appear in the Test results pane.

   The image below shows the data as well as the **Use Dynamic Query Mode** checkbox in the lower left corner, which has been checked.
You can click on the Query Information tab to view the Cognos and Native SQL as well as the XML response from the IBM Cognos BI server.

24. Click **OK**.

You should test all your model objects against the Dynamic Query Mode to ensure that SQL generation is as expected for your requirements. If you are building a DMR model, this includes foundation objects such as Data Source and Model Query Subjects as well as Regular and Measure Dimensions.

Once you have finished building the model, you can create and publish a package that uses the Dynamic Query Mode.

25. In the **Project Viewer**, right-click **Packages**, point to **Create**, and then click **Create Package**.

26. In the **Name** box, type **GOSALES (SQL Server)**, click **Next**, and then click **Finish**.

A prompt appears asking if you wish to open the Publish Wizard.

27. Click **Yes**, deselect **Enable model versioning**, and then click **Next** twice. On the **Publish Wizard - Options** screen, the **Use Dynamic Query Mode** option is available when both of the following conditions apply:

- The **Query Mode** property of the project is set to **Compatible**
- The package contains only supported data sources
28. Select **Use Dynamic Query Mode**, if available.

29. Click **Publish**, and then click **Finish**.

The package is now available in IBM Cognos 10 and will use the Dynamic Query Mode for reports written against this package. In IBM Cognos Connection, the query mode used by the package can be verified in the package properties.

**Figure 61 Package properties showing Query Mode as Dynamic**

---

Create a Project, Connection and Package for NCR Teradata

For a NCR Teradata data source, the IBM Cognos Framework Manager machine requires no additional software if all your Framework Manager projects use the Dynamic Query Mode. If any
of your projects use the Compatible Query Mode, however, a NCR Teradata ODBC driver and its dependant components must be installed and configured on the IBM Cognos BI Framework Manager machine.

In this example a database called \textit{dbc} will be used to illustrate creating a relational data source connection.

1. Open IBM Cognos BI Framework Manager, and then click \textbf{Create a new project}.
2. In the \textbf{Project name} box, type \textit{Teradata DQM Model}.
3. Select \textbf{Use Dynamic Query Mode} if you want all modeling activities and execution of reports based on packages published from the project to use the Dynamic Query Mode, as recommended.
4. Click \textbf{OK}.
   
The Select Languages dialog box appears.
5. Ensure that \textbf{English} is selected, and then click \textbf{OK}.
   
The Metadata Wizard appears.
6. Ensure \textbf{Data Sources} is selected, and then click \textbf{Next}.
7. Click the \textbf{New} button to create a new data source connection.
8. In the \textbf{New Data Source} wizard, click \textbf{Next}, in the \textbf{Name} box, type \textit{dbc(Teradata)}, and then click \textbf{Next}.
9. Under \textbf{Type}, select \textit{Teradata (ODBC)}.
   
   Notice the \textbf{Configure JDBC connection} checkbox. Ensure this box is checked so that information can be provided to connect through the JDBC driver which is required for Dynamic Query Mode.

   \textbf{Figure 62 New Data Source wizard showing Teradata (ODBC) selected and the Configure JDBC connection checkbox}

10. Click \textbf{Next}.
   
   In the next steps, the information provided is based on how the NCR Teradata ODBC driver on the Framework Manager machine and the IBM Cognos BI servers were configured and how security is implemented for NCR Teradata. Connection information and sign on information should be provided by the database administrator.
11. In the \textbf{ODBC Data source} box, type in the ODBC connection name you configured for NCR Teradata, and then under \textbf{Signon}, select the \textbf{Password} check box.
Figure 63 New Data Source Wizard with ODBC data source name provided and the Password box selected

<table>
<thead>
<tr>
<th>Specify the Teradata (ODBC) connection string - New Data Source wizard</th>
</tr>
</thead>
<tbody>
<tr>
<td>Edit the parameters to build a Teradata (ODBC) connection string.</td>
</tr>
<tr>
<td>ODBC data source:</td>
</tr>
<tr>
<td>CompTeradata</td>
</tr>
<tr>
<td>ODBC connect string:</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>Collation sequences:</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>□ Open synchronously</td>
</tr>
</tbody>
</table>

**Timeouts**
Specify the time in seconds, in which you want the database to connect or wait for your reply before timing out.

- **Connect time:** 9
- **Reply times:** 9

**Signon**
Select whether or not authentication is needed, and if so, the type of authentication to use, whether a password is required and whether to create a signon.

- **No authentication**
- **An external signon**:
- **Signon**:
  - **Password**: Yes
  - Create a signon that the Everyone group can use:
    - **User ID**: [Field]
    - **Password**: [Field]
    - **Confirm password**: [Field]

**Testing**

<table>
<thead>
<tr>
<th>Test the connection...</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cancel</td>
</tr>
</tbody>
</table>

12. In the **User ID** box, type in the user ID, in the **Password** and **Confirm password** boxes, type in the password, and then click **Next**.
On the next screen, the JDBC connection information will be provided.
13. In the Server name box, Port number box, and Database name box, enter the values provided by the database administrator.

14. Click Test the connection, and then click Test. On the results page of the connection test, notice the results showing a status of Succeeded for the Dynamic query mode.

15. Click Close, click Close again, and then click Finish.

16. Click Close. The new data source appears in the list and is configured to query using either query mode.

17. Ensure dbc(Teradata) is selected, click Next, expand the database and schema, and then expand Tables.

18. For this database, the following tables will be selected.
   • accts
   • customer
   • trans

19. Click Next, click Import, and then click Finish.

20. In the Project Viewer, expand dbc(Teradata).
The query subjects appear as child objects as shown below.

**Figure 67 Project Viewer showing imported query subjects**

21. Double-click `accts` to open its definition, and then click the **Test** tab.
   
   If the **Query Mode** property of the project is set to **Dynamic** when testing a query subject, the test query will run in Dynamic Query Mode. If the **Query Mode** property is set to **Compatible**, however, there is an option to use the Dynamic Query Mode on the Test tab in the lower left corner, provided that the query subject is for a data source supported by the Dynamic Query Mode.

22. Click the **Use Dynamic Query Mode** check box, if available.

23. Click **Test Sample**.
   
   Framework Manager sends the test query through the IBM Cognos 10 gateway to one of the IBM Cognos BI servers, which, in turn, queries the reporting database. The data retrieved by the test query appear in the **Test results** pane.

   The image below shows the data as well as the Use Dynamic Query Mode checkbox in the lower left corner, which has been checked.

**Figure 68 Query Subject Definition Test tab showing test results**

You can click on the Query Information tab to view the Cognos and Native SQL as well as the XML response from the IBM Cognos BI server.

24. Click **OK**.
You should test all your model objects against the Dynamic Query Mode to ensure that SQL generation is as expected for your requirements. If you are building a DMR model, this includes foundation objects such as Data Source and Model Query Subjects as well as Regular and Measure Dimensions. Once you have finished building the model, you can create and publish a package that uses the Dynamic Query Mode.

25. In the **Project Viewer**, right-click **Packages**, point to **Create**, and then click **Create Package**.
26. In the **Name** box, type **dbc (Teradata)**, click **Next**, and then click **Finish**.
   A prompt appears asking if you wish to open the **Publish Wizard**.
27. Click **Yes**, deselect **Enable model versioning**, and then click **Next** twice.
   On the Publish Wizard - Options screen, when only supported data sources are in the package, there is an option called **Use Dynamic Query Mode**.

**Figure 69 Publish Wizard showing the Use Dynamic Query Mode option**

28. Select **Use Dynamic Query Mode**, if available.
29. Click **Publish**, and then click **Finish**.
   The package is now available in IBM Cognos 10 and will use the Dynamic Query Mode for reports written against this package. In IBM Cognos Connection, the query mode used by the package can be verified in the package properties.
Create a Project, Connection and Package for Oracle

For an Oracle data source, the IBM Cognos Framework Manager machine requires no additional software if all your Framework Manager projects use the Dynamic Query Mode. If any of your projects use the Compatible Query Mode, however, an Oracle client must be installed and configured on the IBM Cognos BI Framework Manager machine.

In this example the sample Great Outdoors Sales Oracle database called GS_DB_ORA will be used to illustrate creating a relational data source connection. This database ships with the product as part of the samples.

1. Open IBM Cognos BI Framework Manager, and then click Create a new project.
2. In the Project name box, type Oracle_DQM_Model.
3. Select Use Dynamic Query Mode if you want all modeling activities and execution of reports based on packages published from the project to use the Dynamic Query Mode, as recommended.
4. Click OK.
   The Select Languages dialog box appears.
5. Ensure that English is selected, and then click OK.
6. The Metadata Wizard appears.
7. Ensure Data Sources is selected, and then click Next.
8. Click the New button to create a new data source connection.
9. In the New Data Source wizard, click Next, in the Name box, type GOSALES(Oracle), and then click Next.
10. Under Type, select Oracle.

Figure 70 Package properties showing Query Mode as Dynamic
Notice the **Configure JDBC connection** checkbox. Ensure this box is checked so that information can be provided to connect through the JDBC driver which is required for Dynamic Query Mode.

**Figure 71 New Data Source wizard showing Oracle selected and the Configure JDBC connection checkbox**

11. Click **Next**.

In the next steps, the information provided is based on how the Oracle clients on the Framework Manager machine and the IBM Cognos 10 servers were configured and how security is implemented for Oracle. Connection information and sign on information should be provided by the database administrator.

12. In the **SQL*Net connect string** box, type the Oracle Service Name defined during the Oracle client configuration, and then under **Signon**, select the **User ID** check box.

**Figure 72 New Data Source Wizard with SQL*Net connect string provided and the Password box selected**
13. In the **User ID** box, type in the user ID, in the **Password** and **Confirm** password boxes, type in the password, and then click **Next**.

   On the next screen, the JDBC connection information will be provided.

   **Figure 73 New Data Source wizard showing JDBC driver parameters**

   ![Specify the Oracle (JDBC) connection string - new Data Source wizard](image)

   You have a choice of which connection type to use. This will be based on how the Oracle clients on each machine have been configured.

14. Select the **Connection Type** setting, depending on the information provided by the database administrator. There are three types:

   - Service ID
   - TNS Names Alias
   - Oracle Net Descriptor

   The **Service ID** option allows you to connect directly to the Oracle database server without an Oracle client. Enter the server name, port number and service ID provided to you by the database administrator.
The **TNS Names Alias** option allows you to connect to the Oracle database through a TNS Name defined in the local Oracle client. Enter the TNS name you defined when configuring the client.

The **Oracle Net Descriptor** option allows you to connect to Oracle using an Oracle Net Connection. Enter the Oracle Net descriptor provided to you by the database administrator. For example:
15. After selecting the desired option and entering the connection information, click **Test the connection**, and then click **Test**. On the results page of the connection test, notice the results showing a status of Succeeded for the Dynamic query mode.

**Figure 77 Test the connection screen showing successful Dynamic query mode test results**

<table>
<thead>
<tr>
<th>Type / Query Mode</th>
<th>Status</th>
<th>Message</th>
</tr>
</thead>
<tbody>
<tr>
<td>Oracle (JDBC)</td>
<td>Dynamic</td>
<td>Succeeded</td>
</tr>
</tbody>
</table>

16. Click **Close**, click **Close** again, and then click **Finish**.
17. Click **Close**. The new data source appears in the list and is configured to query using either query mode.

**Figure 78 Metadata Wizard showing new GOSALES(Oracle) data source**

The next steps will be to import metadata and test query subjects.
18. Ensure **GOSALES(Oracle)** is selected, click **Next**, expand **GOSALESDW**, and then expand **Tables**.

19. Select the following tables, and then click **Next**.
   - **GO_TIME_DIM**
   - **SLS_RTL_DIM**
   - **SLS_SALES_FACT**

20. Click **Import**, and then click **Finish**.

21. In the **Project Viewer**, expand **GOSALES(Oracle)**.
   The query subjects appear as child objects as shown below.

   **Figure 79 Project Viewer showing imported query subjects**

22. Double-click **GO_TIME_DIM** to open its definition, and then click the **Test** tab.
   If the **Query Mode** property of the project is set to **Dynamic** when testing a query subject, the test query will run in Dynamic Query Mode. If the **Query Mode** property is set to **Compatible**, however, there is an option to use the Dynamic Query Mode on the Test tab in the lower left corner, provided that the query subject is for a data source supported by the Dynamic Query Mode.

23. Click the **Use Dynamic Query Mode** check box, if available.

24. Click **Test Sample**.
   Framework Manager sends the request through the IBM Cognos 10 gateway to one of the IBM Cognos BI servers, which generates the query to the reporting database. The data retrieved by the test query appear in the **Test results** pane.

25. The image below shows the data as well as the **Use Dynamic Query Mode** checkbox in the lower left corner, which has been checked.
26. Click **OK**.

You should test all your model objects against the Dynamic Query Mode to ensure that SQL generation is as expected for your requirements. If you are building a DMR model, this includes foundation objects such as Data Source and Model Query Subjects as well as Regular and Measure Dimensions. Once you have finished building the model, you can create and publish a package that uses the Dynamic Query Mode.

27. In the **Project Viewer**, right-click **Packages**, point to **Create**, and then click **Create Package**.

28. In the **Name** box, type `GOSALES (Oracle)`, click **Next**, and then click **Finish**.

A prompt appears asking if you wish to open the **Publish Wizard**.

29. Click **Yes**, deselect **Enable model versioning**, and then click **Next** twice. On the Publish Wizard - Options screen, the Use Dynamic Query Mode option is available when both of the following conditions apply:
   - The **Query Mode** property of the project is set to **Compatible**
   - The package contains only supported data sources
30. Select **Use Dynamic Query Mode**, if available.

31. Click **Publish**, and then click **Finish**. The package is now available in IBM Cognos 10 and will use the Dynamic Query Mode for reports written against this package. In IBM Cognos Connection, the query mode used by the package can be verified in the package properties.

**Figure 82 Package properties showing Query Mode as Dynamic**

IBM Cognos 10 Administration

The Dynamic Query Mode introduces a new query service, named QueryService. IBM Cognos 10 Administration contains new elements to configure, tune, and troubleshoot the query service. Also
available are a command-line API and configuration file settings to manage the cache maintained by the query service. The following sections describe the mechanisms available to administer the query service.

### Status Tab

In IBM Cognos 10 Administration, metrics for the Query Service can be seen on the Status tab under System. Navigate to the QueryService service in the Scorecard pane to view metrics as well as Logging and Tuning settings.

**Figure 83 IBM Cognos Administration - Status tab - System section showing the Metrics and Settings panes for the query service**

The Scorecard pane indicates which servers, dispatchers, and services are available and allows for administrative tasks such as starting and stopping the service or setting properties.

The Metrics pane displays statistics and just as with other services, certain metrics have configurable thresholds. They are edited by clicking the Edit icon (pencil) to the right of each metric.

The Settings pane indicates how the selected item in the Scorecard pane is configured. The Logging and Tuning settings can be edited in this section as well by clicking on the Set properties icon in the top right corner of the Settings pane. They can also be edited in the Dispatchers and Services section under the Configuration tab, which is discussed in the next section.

### Configuration Tab

On the Configuration tab, there are four locations pertaining to the Dynamic Query Service:

1. **Data Source Connections** for configuring data sources including supported Dynamic Query Mode data sources
2. **Content Administration** for scheduling Query service administration tasks
3. **Dispatchers and Services** for configuring the QueryService service
4. **Query Service Caching** to immediately perform cache tasks

**Data Source Connections**

You can create data sources for IBM Cognos 10 in **Data Source Connections** under the Configuration tab.

**Figure 84 IBM Cognos Administration - Configuration tab - Data Source Connections**

Supported data sources for Dynamic Query Mode will indicate a successful connection through the Dynamic Query service when testing the connection. To verify success, look for **Succeeded** in the **Status** column for the entry with **Query Mode** value **Dynamic**, as shown in the following image

**Figure 85 Result of testing the connection, showing success for both Compatible and Dynamic Query Modes**

**Content Administration**

Content Administration under the Configuration tab has a **New Query Service Administration Task** button as shown in the image below.
Clicking the button displays a pop-up menu to select the data source type. Selecting one of the available types launches the **Query Service Administration Task** wizard. The wizard guides you through the steps to configure and schedule the task.

On the first page of the wizard, enter a name for the new task. You can also supply a description and screen tip for the task. The next page prompts you to select the options for this task, which are dependent on the data source type that you selected in the initial pop-up menu.

If you selected **Dimensionally-modeled relational** as the data source type, the options page appears as shown in the image below.

**Figure 87 New Query Service Administration Task wizard - Select the options page for a dimensionally-modeled relational data source**

On this page, you provide values for two mandatory fields, depending on which **Type** of object you select:

- **Operation** – either **Clear cache** or **Write cache** state, as explained below for an OLAP data source type
• **Package** – name of the package to which the task applies, if you select the **Package** object type
• **Data source** – name of the data source to which the task applies, if you select the **Data source** object type

If you selected an OLAP data source type (anything other than **Dimensionally-modeled relational**), the options page appears as shown in the image below.

**Figure 88 New Query Service Administration Task Wizard – Select the options page for an OLAP data source**

The four fields on this page are mandatory:

• **Operation** – either **Clear cache** or **Write cache state**, as explained below
• **Datasource** – the name of the data source to which the task applies
• **Catalog** – a provider-dependent catalog name within the data source
• **Cube** – name of the OLAP cube within the specified catalog

There are two query service administration operations from which to choose:

• **Clear cache** - Clear the Dynamic Query cache to avoid using outdated data
• **Write cache state** - Write the cache state to a file for cache use analysis

You must also specify a single data source, catalog and cube name, or use an asterisk (*) to apply the task to all items of that type. For example, to write a cache state file for an Oracle Essbase cube, the settings might be as follows (and shown in the image below):

• **Operation**: Write cache state
• **Datasource**: Essbase
• **Catalog**: GODB
• **Cube**: GODB
Figure 89 Example Settings to Write Cache State for an Oracle Essbase Cube

In this case, the cube and catalog name are the same. In general, you can find the correct syntax to enter in the wizard by following these steps:

1. Run a report against the data source in question.
2. Use the manual Write cache state feature (described under Query Service Caching later in this chapter) to capture the cache state for all caches.
3. In that file, look for the required entries such as Data source, Catalog, and Cube for the data source.
4. The values in these entries are the ones you need for the corresponding fields of the wizard.

You can schedule a query service administration task in the same way as any other administration task.

Note: If you schedule any Clear cache tasks to run at pre-determined times, you should selectively disable automatic cache clearing as described under Automatic Cache Clearing.

These administration tasks affect all server groups in a distributed environment. In other words, depending on the task run, each IBM Cognos 10 report server will either have its cache cleared for the specified data source, or have the cache state file written to its local logs/XQE directory.

For IBM Cognos TM1, there is no caching on the IBM Cognos 10 side therefore these tasks do not apply to that data source.

The Write cache state operation creates a time-stamped XML file showing the state of specified caches, which allows you to verify that caches are being cleared. It may also be useful for troubleshooting purposes under the guidance of IBM Cognos development. The XML file is written to the c10\logs\XQE\ directory and has a filename of the form SALDump_<datasourceName>_<catalogName>_<cubeName>_timestamp.xml, for example, SALDump_Essbase_GODB_GODB_1281624776529.xml.

Sample file output:

```xml
<?xml version="1.0" encoding="UTF-8" ?>
<xqeCacheMetric>
<dataSource type="EB">
  <Essbase>
```

IBM Business Analytics Proven Practices: The Cognos BI 10.1.1
Dynamic Query Cookbook
The **Write cache state** feature can validate cache clearing. You can tell that a cache has been cleared if the XML content of the file indicates a failure to match the names that you specify in the wizard, for example:

```xml
<?xml version="1.0" encoding="UTF-8" ?>
<xqeCacheMetric>No cached cube found matching the criteria: dataSource name = Essbase catalog name = GODB cube name = GODB</xqeCacheMetric>
```

**Dispatchers and Services**

In Dispatchers and Services under the Configuration tab, there is now a QueryService item which is used to configure settings for the Dynamic Query Mode.

**Figure 90 IBM Cognos Administration - Configuration tab - Dispatchers and Services highlighting the QueryService item**

![Configuration tab with QueryService item highlighted](image)

The query service has a number of settings grouped in three categories:
• Environment
  • Advanced settings

• Logging
  • Audit logging level
  • Enable query execution trace?
  • Enable query planning trace?

• Tuning
  • Write model to file?
  • Additional advance settings for the Java Virtual Machine (JVM)

By default, an instance of the query service acquires its configuration settings from its parent. You can override the acquired values by setting them explicitly on the **Settings** tab of the Set properties screen for the **QueryService** item, as shown in the following image.

**Figure 91 Set properties screen for the query service**

![Set properties screen for the query service](image)

The Advanced settings, Audit logging level, and Idle connection timeout are inherited standard settings for all services that may or may not apply to QueryService. Advanced settings allow for additional service settings provided by IBM for specific and typically less common scenarios.

**Enable query execution trace?**

Enables or disables a query execution trace. To enable it, select the checkbox in the **Value** column and click the **OK** button. The trace configuration change will take effect within 15 seconds.

Enabling the query execution trace setting will write information such as the native MDX to a run tree log within the $c10/logs/XQE$ directory. Profiler information is also written to capture execution and waiting time metrics for query constructs. Since a log is generated for each report that is executed, the log file adheres to the following naming convention:

- `timestamp_reportName/runtreeLog.xml`
- `timestamp_reportName/profilingLog-#.xml`
As an example, executing a report called top_sales would result in a log file named 2010-05-10_11h33m700s_top_sales/runTreeLog.xml and one or several profiler logs named 2010-09-10_11h33m700s_top_sales/profilingLog-0.xml, 2010-09-10_11h33m700s_top_sales/profilingLog-1.xml, ...etc.

Some report execution requires executing sub-queries. Sub-queries execution trace is stored under a separate directory within the main report directory. The sub-query directory contain the same logging elements as the main report, runTreeLog.xml and profilingLog-#.xml.

If executing the report top_sales requires the execution of one or more sub-queries, the trace for those sub-queries is stored in 2010-09-10_11h33m700s_top_sales/subqueries.

These files can be analyzed with the Dynamic Query Analyzer which is covered in Section 9.4 of this book.

**Enable query planning trace?**

Enables or disables query plan tracing, also known as a plan tree, which captures the transformation process of a query. To enable it, select the checkbox in the Value column and click the OK button. The trace configuration change will take effect within 15 seconds.

You can use this information for advanced understanding of the decisions and rules that are executed to produce an execution tree. The query planning trace is logged for every query that is executed using Dynamic Query Mode. The planning trace logs are located on the report server servicing the request in the c10\logs\XQE\ directory with the following naming conventions.

- timestamp_reportName/planningLog.xml
- timestamp_reportName/planningLog_pass_###.xml

Since planning logs are large, there may be an impact on query performance when this setting is enabled.

**Idle connection timeout**

Specifies the number of seconds to maintain an idle data source connection for re-use. The default setting is 300. Valid entries are 0 to 65535. Lower settings reduce the number of connections at the expense of performance. Higher settings may improve performance but raise the number of connections to the data source.

**Write model to file?**

Specifies if the query service should write the model to a file when a query is executed. The file is used only for troubleshooting purposes, under the guidance of Customer Support. The file is given the name c10\logs\model\packageName.txt.

**Additional JVM arguments for the query service**
This option allows you to pass additional parameters to the DQM JVM instance. This setting should only be used under the guidance of Customer Support.

**Initial JVM heap size for the query service (MB)**

This setting defines how much memory the DQM Java Virtual Machine will take on startup. The value set here is passed to the JVM on startup as the –Xms<value> parameter.

**JVM heap size limit for the query service (MB)**

This setting defines the upper memory limit of the DQM Java Virtual Machine during operation. The value set here is passed to the JVM on startup as the –Xmx<value> parameter.

**Query Service Caching**

It may be necessary to clear the cache manually if the data source data changes infrequently or if it is required to clear the cache in between automatically scheduled cache clearing.

The **Query Service Caching** section under the Configuration tab allows for manual Dynamic Query cache clearing and writing the cache state to file for one or more server groups. Select **Query Service Caching** in the vertical menu if you want to initiate operations on the query service cache.

**Figure 92 IBM Cognos Administration - Configuration tab - Query Service Caching**

The **Write cache state** feature creates a time-stamped XML file (with filename \c10\logs\XQE \SALDump_all_all_all_timestamp.xml) showing the state of all caches. In a distributed installation, each report server that has a cache will write the cache state file to its local logs directory.

**Command-line API**

In addition to the administration interface for executing and scheduling cache management tasks, there is a command-line API that enables manual and automated cache management outside the normal IBM Cognos 10 administration environment. The command-line utility is located in the c10\bin directory and is called QueryServiceAdminTask.sh or QueryServiceAdminTask.bat depending on your operating system.

The **QueryServiceAdminTask** utility accepts one or two arguments:
1. Cache operation (mandatory)
2. Cache subset (optional)

For the first argument, specify one of the following values to select the corresponding cache operation:

- 1 – Clear cache
- 2 – Write cache state

Use the second argument to specify the portion of the cache to which the cache operation applies, by naming a data source, catalog, and cube (separated by forward slash characters). You can use the wildcard character (*) to represent all data source, catalog, or cube names. Omitting this argument causes the cache operation to apply to the entire cache.

For example, to clear the cache for all cubes in all catalogs under all data sources, enter the following command in a command shell:

```
queryServiceAdminTask 1 "*/******/"
```

or just

```
queryServiceAdminTask 1
```

Entering `queryServiceAdminTask -help` in a command shell displays detailed usage instructions for the utility.

Because this command-line utility makes an immediate task request, it does not go through the job scheduler and monitoring service. Consequently, it only affects the IBM Cognos 10 server on which the command is run.

**Automatic Cache Clearing**

By default, the Dynamic Query Mode clears its caches every 24 hours to avoid producing report output based on potentially stale data. The actual time at which each cache is cleared depends on when the cache was created.

You can override the default interval for automatic cache clearing or disable it entirely for each type of data source (for example, SAP BW, Essbase, DMR) and for specific data sources, catalogs and cubes. To accomplish this, edit the configuration file `c10/configuration\xqe.securecache.config.xml`, which contains comments describing each parameter, possible values and their interpretation.

If you create any scheduled jobs to clear the Dynamic Query caches for any combination of data sources, catalogs and cubes, you should disable automatic cache clearing for the corresponding data sets.

After restarting the IBM Cognos 10 service, changes to this file affect cache clearing only for instances of the `QueryService` running on the server where the configuration file resides. In a distributed environment, you must make these changes on each server.
IBM Cognos 10 Caching

While previous releases of IBM Cognos BI have some level of caching, IBM Cognos 10 Dynamic Query Mode provides a greater degree of secure, smart caching offering significant performance improvements for most queries and workloads.

The main purpose of the cache is to leverage previously executed results for reuse, thereby avoid roundtrips to the database whenever possible. The performance benefits of the cache will be most noticeable when executing:

- similar reports with small modifications
- repeated analysis within the same cube
- repetitive master-detail requests for large reports

Cache loading occurs on-demand, that is, as requests are received and executed.

Online Analytical Processing (OLAP) Cache

IBM Cognos BI versions 8.2 through 8.4 leveraged a metadata/data fetch concept when querying certain data sources. With the level of complexity of reports currently being authored, a greater local processing control yielded performance benefits. Alternatively, some data sources still leveraged a remote type of approach where processing was pushed down to the respective databases. Only the local processing approach leverages the secured cache.

The principle behind the local processing approach is to retrieve the raw data from the underlying data source and process everything else locally. Some level of aggregation, filtering, and other simple functions may still be sent to the data source. The local approach avoids pitfalls occasionally encountered when trying to push complex native SQL/MDX to the querying data source, which can incur performance penalties.

The local approach to OLAP reporting is broken down into two simple steps: metadata fetch and data fetch. When a report is executed, we initially retrieve all members requested (metadata), either by level and/or unique member inclusion, and then utilize the retrieved members in order to construct the MDX used for data retrieval (facts). As these calls are performed, for both metadata and data requests, each result returned is cached and can potentially be reutilized when further requests are made within the same context.

The concept of context is important to understand in the realm of the secured cache. It identifies a result based on what was requested. For example, the context will be built using who the query was executed by, which cube was queried, which particular year was used for filtering, etc. In essence, anything that narrows the scope of a result which is also sent to the data source will be considered as context. If any subsequent requests are made within the same context, the cached results will be used. The context may affect both metadata and data cache.

SAP BW data sources can still leverage remote processing. This method relies on the underlying data source to process the entire MDX request (minus some exceptions). This capability can
be leveraged by changing the query processing to “Database Only” as explained in the next paragraph. Some reports lend themselves very well to this type of processing – for example simple grouped list reports such as target reports for drill-through. In these scenarios, since the report complexity is greatly reduced, the entire request will benefit from avoiding the metadata fetch portion, simply retrieving the data and presenting it to the user. However, this result will not leverage the cache and cannot be reused.

In order to force the behaviour of a particular mode (local vs. remote) on a query-by-query basis, you can change a query hint in Report Studio, as shown in the image below:

1. Select the query in question.
2. In the left hand pane, look in the pane titled Properties - Query. Under the Query Hints section, select the desired setting for the Processing property:
   - (Default) follows the server’s default behaviour
   - Database only forces remote behaviour
   - Limited Local forces local processing behaviour

![Figure 93 IBM Cognos Report Studio query properties showing the available processing options](image)

This change is only possible in Report Studio at this time. All other studios use the default, local processing.

A Practical OLAP Cache Example

Consider a report designed to retrieve all product sales for all customers within each sales region for 2009. The report author creates a list report with three nested dimensions – Sales Region, Customer, and Product. There are two measures of interest, Quantity purchased and Unit Price. Only one year’s results are pertinent, so the author adds a slicer for Year 2009. The following image shows the resulting report layout, which will display the two measures for each Product, grouped first by Customer and then by Sales Region.
Figure 94 List report for product sales by customer and region

![Figure 94 List report for product sales by customer and region](image)

When this report is executed, the Dynamic Query Mode fetches all members for Sales Region, Customer, and Product, storing the results in the cache. The image below illustrates the one-to-one relationship between report columns and sections of the metadata cache.

Figure 95 Relationship between report columns and sections of the Metadata Cache

![Figure 95 Relationship between report columns and sections of the Metadata Cache](image)

In this example, the cache contains the following members in three sections:

- Sales Region: North, South, East, West
- Customer: BobShop, BTTools, GTGarage, AutoZone
- Product: Wipers, Bumpers, Mirrors, Oil

The query transformation engine uses the enumerated members to construct an MDX statement to retrieve the nested members with the requested measures. The following image illustrates the generation of an MDX statement.
Figure 96 MDX query generated from members enumerated in the metadata cache

For this report, the generated MDX is a `SELECT` statement involving a cross-join of the requested measures (Quantity and Unit Price) with the enumerated members of Sales region, Customer, and Product for the Year 2009.

The query execution engine sends the simplified constructed MDX to the data source to retrieve the fact data. The result set is stored in the data cache to subsequently produce the desired output, as illustrated in the following images.

Figure 97 Metadata cache, generated MDX and data cache containing data set resulting from executing the generated MDX
Two phases took place here, the metadata phase to extract all requested members, and a second phase fetching those members with fact data.

Suppose that the report author now adds a calculation such as Quantity x Unit Price to the report as a new column. Given that the calculation involves the same members and measures as before, the Dynamic Query Mode can reuse the metadata and data from the previous request to produce the desired output, thus bypassing the data source. This optimization is possible because calculations are processed locally and no new metadata or data needs to be fetched from the database.
Alternatively, if the user then decides to add another measure from the data source, we will be able to reuse the metadata (first phase) but will need to re-execute the data fetch (second phase) given a new measure has been added.

While this example oversimplifies the cache, the demonstrated principles of operation remain valid.

**Context Dependency**

The cache is context-dependent. In the example given above, the year, although not included/ projected in the report, serves as a context as it is used in a slicer. The data retrieved is in context of year 2009. Filtering on a different year or adding another filter would offer a new cache context at which time the previous cache may not be reusable.

Another important context is the user. Each data source can secure its data based on proprietary authorization rules and techniques. Therefore, user X may be entitled to view all data while user Y may be limited to view a subset of the data. As a result, each user id becomes an element of the context and metadata/data cannot be shared among users, thereby keeping the cache secure in a multiuser environment.

**DMR Cube Cache**

For OLAP reporting over relational data sources, the Dynamic Query Mode creates and uses DMR cube caches to minimize repeated requests to the relational database for metadata and data values. Each cube cache is associated with a single DMR package and consists of two sub-caches:

- Metadata cache containing members and properties
- Cell-value cache containing data values

A DMR package could have multiple associated cube caches under any of the following conditions:

- Multiple versions of a package in use at the same time
- A package accessed by multiple users having different roles
- Queries involving different pre-aggregation detail filters (including model filters)
- Queries involving different slicers

**Cache Re-use**

When a DMR package has multiple cube caches, the metadata cache is shared by all its associated cube caches by applying the appropriate security context to the metadata. In other words, there is a single metadata cache per DMR package.

Each cube cache has its own independent cell-value cache. However, multiple queries involving the same package re-use an existing cache whenever possible. For example, if the same detail filter or slicer appears in multiple reports on the same package, the cube cache is re-used instead of creating a new one each time.
There is no sharing of caches between packages, even for packages created from the same model.

**Populating the Cube Cache**

The Dynamic Query Mode populates cube caches on-demand, as metadata and cell values are requested.

It is possible to pre-populate a cube cache by executing a simple list report. The administrator could schedule the report for execution following a scheduled task to clear the cache, if desired. See the previous section on Content Administration for information on scheduling a task to clear the cache.

**Disabling Cube Caching**

It is possible to disable cube caching for all DMR packages on a server by modifying the configuration file `c10\configuration\xqe\dmr.properties` directory. To do this, change the value of the `enableDMRCubeReuse` parameter from `true` (default) to `false`.

**Performance Considerations for Caching**

Not all data requests will be cached. Some types of requests/results would not benefit from being cached. For example, very large batch type reports which results in hundreds or thousands of pages would spend more time writing results to the cache then it would take re-executing the same request subsequent times. As well, some queries perform well when executed on the underlying data source and may not benefit from using the cache. The Dynamic Query Mode automatically determines when to avoid writing results to the cache for queries that would not benefit from using it.

When using the cache, the Query Execution engine caches each result in the context of all dimensions in the published package. While many factors affect read/write performance from the cache, having a high number of dimensions will have a negative impact on cache performance. Therefore, it is a good practice to limit the choice of dimensions in a package to those actually required to satisfy the business requirements. This will give a noticeable performance gain for some situations.

**Query Semantics for OLAP over Relational**

When using a dimensionally modeled relational (DMR) package, Dynamic Query Mode provides a true OLAP-over-relational experience that is consistent with pure dimensional (OLAP) data sources. This consistency shows up in various aspects of query semantics:

- Member ordering
- Null suppression
- Nulls in calculations
Member Ordering

Order of members is a fundamental aspect of dimensional analysis. Member-relative and time-series functions such as `NextMember`, `PreviousMember`, `ClosingPeriod` and `OpeningPeriod` rely on the order of members to provide meaningful results.

Dynamic Query provides two mechanisms to ensure consistent ordering of members:

- Natural order
- Sort specification

Natural Order

The default order of members is called the natural order. The natural order of members in OLAP over relational is ascending order by the member caption (with nulls last). In the case of a collision on caption, the secondary sort key is the business key in ascending order. The business key is assumed to be unique.

Consider a Return Reason level in a dimension of a data model for product sales. In the absence of explicit member sort specifications in the model, Dynamic Query Mode sorts the members in alphabetical order according to the member caption, for example:

1. Defective product
2. Incomplete product
3. Unsatisfactory product
4. Wrong product ordered
5. Wrong product shipped

In Compatible Query Mode, by contrast, there is no consistent default sort and the natural order is dependent on the query and the data source.

Sort Specification

IBM Cognos Framework Manager provides the option to explicitly set the member sort specification according to your business needs. This option is available through Dimension Definition dialog on the Members Sort tab, as shown in the following image.
Member-relative analysis always respects the order of members, whether explicit or implicit (natural order). If a particular order of the members is important to the business view, then use the Level Sort Properties of this dialog to define an explicit sort specification that reflects the business view of such members. For more information on how to specify member sort order, see the IBM Cognos 10 Framework Manager User Guide.

Multiple Level Sort Properties apply in the order listed. In the case of collision for all sort properties, the natural order applies.

When there is no explicit sort specification, the natural order will prevail. Relying on the natural order can cause member-relative functions to give unexpected results when a particular order of members is required, notably for the month level of time dimensions.

The Dynamic Query Mode ignores the Sorting Options that appear on the Members Sort tab. Instead, it applies the semantics shown in the following table.
Table 5 Sorting options and the effect on the Dynamic Query Mode

<table>
<thead>
<tr>
<th>Option</th>
<th>Semantics for Dynamic Query Mode</th>
</tr>
</thead>
<tbody>
<tr>
<td>Metadata</td>
<td>The metadata tree display always respects the order of members, whether explicit or implicit (natural order).</td>
</tr>
<tr>
<td>Data</td>
<td>The order of members, whether explicit or implicit (natural order) is the default ordering for report output. The report author can apply a different sort for display of members.</td>
</tr>
</tbody>
</table>

Member Ordering vs. Report Sorting

Remember that member ordering defined in the model follows the hierarchy defined for the dimension. However, a report sort follows the report layout. For example, consider a Retailers dimension, which defines a hierarchy of four levels:

1. Region
2. Retailer country
3. Retailer
4. Retailer site

Assume that, in the model, the Retailer country level has an ascending sort on the member caption, which contains the country name. When projecting the Retailer country level in a report, the members appear in order by caption within Region, according to the defined hierarchy. The output resulting from projecting Region and Retailer country in a list report (without report sorting) might look like the following report output.

Figure 101 IBM Cognos Report Viewer output of a list of Region and Retailer country

<table>
<thead>
<tr>
<th>Region</th>
<th>Retailer country</th>
</tr>
</thead>
<tbody>
<tr>
<td>Americas</td>
<td>Brazil</td>
</tr>
<tr>
<td>Americas</td>
<td>Canada</td>
</tr>
<tr>
<td>Americas</td>
<td>Mexico</td>
</tr>
<tr>
<td>Americas</td>
<td>United States</td>
</tr>
<tr>
<td>Asia Pacific</td>
<td>Australia</td>
</tr>
<tr>
<td>Asia Pacific</td>
<td>China</td>
</tr>
<tr>
<td>Asia Pacific</td>
<td>Japan</td>
</tr>
<tr>
<td>Asia Pacific</td>
<td>Korea</td>
</tr>
<tr>
<td>Asia Pacific</td>
<td>Singapore</td>
</tr>
<tr>
<td>Central Europe</td>
<td>Belgium</td>
</tr>
<tr>
<td>Central Europe</td>
<td>France</td>
</tr>
</tbody>
</table>

Even without projecting Region, the order of members in the Retailer country column would remain the same. If you want to apply an alphabetical sort on Retailer country that doesn’t follow the hierarchical structure of the dimension, apply a sort to the column in the report layout. For the same data, the output from projecting just Retailer country in a list report with an alphabetical sort applied to the column would be as shown in the following report output.
The report sort doesn’t interfere with any member-relative operations, which continue to obey the member sort order (explicit or implicit) in the model.

### Null Suppression

The Dynamic Query Mode brings null suppression behaviour for OLAP-style reporting over relational data sources (based on a DMR package) into alignment with the behaviour for other OLAP data sources. The implications of this improved consistency are best described in terms of the **Suppress** query property and related behaviour in the Compatible and Dynamic Query Modes.

Figure 103 shows the **Properties - Query** pane of Report Studio with a property called **Suppress** that can have one of three states:

- **(Default)**, when the property value is not set, implying data-provider-dependent behaviour with respect to null suppression
- **None**, implying that no values are suppressed from report output
- **Nulls**, implying that null values are suppressed

In Compatible Query Mode, however, the **Suppress** query property is ignored for a report based on a DMR package. Instead, null suppression will occur for all hierarchies in a given dimension if and only if all the following conditions are met:
• There is a measure, a measure calculation, or at least two dimensions in the query
• The fact table behind the measure does not include rows with null fact values
• The dimension does not have OLAP-compatible ordering enabled in the model

You set OLAP-compatible sorting in the Members Sort tab of the Dimension Definition dialog box of IBM Cognos 10 Framework Manager, as shown in the following image. In the Sorting options area, the Data option and sub-option Always (OLAP compatible) have been selected and applied to the selected Time dimension.

**Figure 104 - Members Sort tab of the Dimension Definition dialog with OLAP-Compatible sorting selected**

The following image shows the Members Sort properties of a time dimension that does not have OLAP-compatible sorting. No Sorting options are set for the Time (close date) dimension. Therefore, it does not have OLAP-compatible ordering.
In Dynamic Query Mode, the Suppress query property is respected for all data sources. When the Suppress property is not set (Default), the default behaviour for DMR is to suppress nulls, as if the Suppress property were set to Nulls. In addition, the model setting for Sorting options is ignored.

In most scenarios, end users want the improved report visualization that comes with null suppression. It can also improve performance, particularly with very sparse data.

The following table shows when null suppression occurs for the various combinations of the Suppress query property, Model Ordering and Query Mode (Compatible and Dynamic). Each of these combinations will be further explained through an example.

**Table 6 Contrasting the effect of the Suppress and Mode Ordering on both query modes**

<table>
<thead>
<tr>
<th>Example #</th>
<th>Suppress Property</th>
<th>Model Ordering</th>
<th>Compatible Query Mode</th>
<th>Dynamic Query Mode</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>None</td>
<td>None</td>
<td>Nulls suppressed</td>
<td>No suppression</td>
</tr>
<tr>
<td>2</td>
<td>None</td>
<td>OLAP</td>
<td>No suppression</td>
<td>No suppression</td>
</tr>
<tr>
<td>3</td>
<td>Nulls or (Default)</td>
<td>None</td>
<td>Nulls suppressed</td>
<td>Nulls suppressed</td>
</tr>
<tr>
<td>4</td>
<td>Nulls or (Default)</td>
<td>OLAP</td>
<td>No suppression</td>
<td>Nulls suppressed</td>
</tr>
</tbody>
</table>

The following examples illustrate the behaviour of the Compatible and Dynamic Query modes for the four combinations of the Suppress property and model ordering. There is no difference in the behaviour between the modes for examples 2 and 3. They are included here for completeness.
All examples involve crosstab reports created in IBM Cognos Report Studio. The following image shows a typical crosstab report layout, with the Quantity measure in the cells, Mountaineering Equipment from Product line on the column edge, and Year from a time dimension on the row edge.

**Figure 106 - Typical report layout for the following examples**

In order to illustrate the effect of sort order in the Compatible Query Mode, the examples use two different time dimensions:

- **Time**, which has OLAP-compatible sorting set in the model
- **Time (close date)**, which does not have sorting set in the model

These two dimensions have different member captions and present different views of the data. Therefore, actual report layouts and the report outputs below differ in detail depending on which time dimension is involved. However, the results are comparable with respect to null suppression.

**Example 1 - Suppress None, No Model Ordering**

This example uses the Time (close date) dimension. The Suppress query property is set to None. This is the default in Analysis studio when General Suppression is not applied.

In Compatible Query Mode, the Suppress query property is ignored and the nulls are suppressed. The following image shows the resulting report output, which includes only years 2005, 2006 and 2007, for which the data values are not null.

**Figure 107 IBM Cognos Report Viewer output of a crosstab displaying no Null values**

<table>
<thead>
<tr>
<th>Year</th>
<th>Mountaineering Equipment</th>
</tr>
</thead>
<tbody>
<tr>
<td>2005</td>
<td>2,571,920</td>
</tr>
<tr>
<td>2006</td>
<td>3,703,840</td>
</tr>
<tr>
<td>2007</td>
<td>3,624,331</td>
</tr>
</tbody>
</table>

By contrast, Dynamic Query Mode obeys the Suppress property (None) and displays the nulls. The following output shows the quantity of mountaineering equipment sold for the years 2004 through to 2007. The data value for 2004 is null.

**Figure 108 IBM Cognos Report Viewer output of a crosstab displaying the Null value for 2004**

<table>
<thead>
<tr>
<th>Year</th>
<th>Mountaineering Equipment</th>
</tr>
</thead>
<tbody>
<tr>
<td>2004</td>
<td></td>
</tr>
<tr>
<td>2005</td>
<td>2,571,920</td>
</tr>
<tr>
<td>2006</td>
<td>3,703,840</td>
</tr>
<tr>
<td>2007</td>
<td>3,624,331</td>
</tr>
</tbody>
</table>
If the Report Author wants to obtain the same output from the Dynamic Query Mode as that produced by the Compatible Query Mode, there are two possible actions:

- Set the **Suppress** property to **Nulls**.
- Use one of the options of the **Suppress** command (**Data > Suppress**) or the **Suppress** button.

The following image shows the **Suppress** drop-down menu for a crosstab report in Report Studio with the **Suppress Rows and Columns** option selected.

**Figure 109 - Suppress drop-down menu with Suppress Rows and Columns selected**

![Suppress drop-down menu](image)

**Example 2 - Suppress None, OLAP-compatible Ordering**

Although this example exhibits no difference in behaviour between the Compatible and Dynamic Query modes, it is included here for completeness. The report involves the **Time** dimension, which has OLAP-compatible ordering specified in the model. The Suppress property is set to **None** as in Example 1.

In both Compatible and Dynamic query modes, the nulls are displayed. The following output shows the quantity of mountaineering equipment sold for the years 2004 through to 2007. The data for 2004 is null.

**Figure 110 IBM Cognos Report Viewer output of a crosstab displaying the Null value for 2004**

<table>
<thead>
<tr>
<th>Quantity</th>
<th>Mountaineering Equipment</th>
</tr>
</thead>
<tbody>
<tr>
<td>2004</td>
<td></td>
</tr>
<tr>
<td>2005</td>
<td>2,571,620</td>
</tr>
<tr>
<td>2006</td>
<td>3,700,840</td>
</tr>
<tr>
<td>2007</td>
<td>3,624,331</td>
</tr>
</tbody>
</table>

**Example 3 - Suppress Nulls, No Model Ordering**

Although this example exhibits no difference in behaviour between the Compatible and Dynamic Query modes, it is included here for completeness. The report uses the same **Time (close date)** dimension as Example 1, but with the **Suppress** query property set to **Nulls** (or not set).

In both Compatible and Dynamic Query Modes, the nulls are suppressed. The following output shows the resulting report output, which includes only years 2005, 2006 and 2007, for which the data values are not null.
**Figure 111 IBM Cognos Report Viewer output of a crosstab displaying no Null values**

<table>
<thead>
<tr>
<th>Quantity</th>
<th>Mountaineering Equipment</th>
</tr>
</thead>
<tbody>
<tr>
<td>2005</td>
<td>2,571,920</td>
</tr>
<tr>
<td>2006</td>
<td>3,703,840</td>
</tr>
<tr>
<td>2007</td>
<td>3,624,331</td>
</tr>
</tbody>
</table>

**Example 4 - Suppress Nulls, OLAP-compatible Ordering**

This example involves the Time dimension, which has OLAP-compatible ordering specified in the model. The Suppress query property is set to Nulls (or not set).

In Compatible query mode, the nulls are displayed. The result is the same as in Example 2. The following table shows the quantity of mountaineering equipment sold for the years 2004 through to 2007. The data for 2004 is null.

**Figure 112 IBM Cognos Report Viewer output of a crosstab displaying the Null value for 2004**

<table>
<thead>
<tr>
<th>Quantity</th>
<th>Mountaineering Equipment</th>
</tr>
</thead>
<tbody>
<tr>
<td>2004</td>
<td></td>
</tr>
<tr>
<td>2005</td>
<td>2,571,920</td>
</tr>
<tr>
<td>2006</td>
<td>3,703,840</td>
</tr>
<tr>
<td>2007</td>
<td>3,624,331</td>
</tr>
</tbody>
</table>

By contrast, the Dynamic Query Mode obeys the Suppress query property and suppresses the nulls. The following table shows the resulting report output, which includes only years 2005, 2006 and 2007, for which the data values are not null.

**Figure 113 IBM Cognos Report Viewer output of a crosstab displaying no Null values**

<table>
<thead>
<tr>
<th>Quantity</th>
<th>Mountaineering Equipment</th>
</tr>
</thead>
<tbody>
<tr>
<td>2005</td>
<td>2,571,920</td>
</tr>
<tr>
<td>2006</td>
<td>3,703,840</td>
</tr>
<tr>
<td>2007</td>
<td>3,624,331</td>
</tr>
</tbody>
</table>

If the Report Author wants to obtain the same output from the Dynamic Query Mode as that produced by the Compatible Query Mode, the recommended action is to set the Suppress query property to None.

**Null Suppression in IBM Cognos BI Studios**

In Report Studio, the Dynamic Query Mode respects the Suppress query property. When set to Default, null suppression occurs. For Compatible Query Mode, the Suppress query property is ignored and the model ordering is followed.

In Analysis Studio, the Suppress query property is set explicitly to None if report Row and Column suppression is not used. If Row and Column suppression is set, the Suppress property is not set.
In Query Studio, you cannot set the **Suppress** query property.

### Treat Nulls as Zeros within Calculations

By default, the Dynamic Query Mode propagates nulls in calculations. However, it is possible to override the default behaviour for DMR packages through configuration settings similar to those for pure OLAP data sources.

**Impacts:** Calculations on data items that contain null data values.

**Usage:** This set of parameters controls whether or not null data values are treated as zeros when used in calculations. If the parameters are enabled, 100 + null would result in 100. If the parameters are disabled, 100 + null would result in null. By default, these parameters are disabled.

**Interoperability with other parameters:** None

**Setting these parameters:** The parameters are available within the file `c10\configuration\xqe\dmr.properties` as shown here (with the default settings):

```properties
null.plus.operator=null
null.minus.operator=null
null.multiply.operator=null
null.divide.numerator=null
null.divide.denominator=null
null.modulo.dividend=null
null.modulo.divisor=null
```

To enable this feature, change the null values to zero as follows:

```properties
null.plus.operator=zero
null.minus.operator=zero
null.multiply.operator=zero
null.divide.numerator=zero
null.divide.denominator=zero
null.modulo.dividend=zero
null.modulo.divisor=zero
```

These changes will be picked up once the IBM Cognos 10 service is restarted. After the restart, this change will affect all queries against any DMR data source through IBM Cognos 10. In a distributed environment, this change will need to be made on all IBM Cognos 10 servers performing data access.

### Troubleshooting the Dynamic Query Mode

The Dynamic Query Mode offers several tracing capabilities that can help in troubleshooting query-related issues. Trace settings for the Dynamic Query Mode are accessible through the IBM Cognos Administration portal via the properties of the QueryService service. By default, the trace files are written to the `c10\logs\XQE` directory, where the c10 represents the IBM Cognos 10 install directory. However, the trace output directory can be configured through a configuration file change.
**Query Execution Trace**

Enabling the query execution trace will write information such as the native MDX to a run tree log within the `c10\logs\XQE` directory, where the `c10` represents the IBM Cognos 10 install directory. Profile information is also written to capture execution and waiting-time metrics for query constructs. Since a log is generated for each report that is executed, the log file adheres to the following naming convention.

```
timestamp_reportName/runtreeLog.xml
timestamp_reportName/profilingLog-#.xml
```

As an example, executing a report called “Retailers” would result in a log file named `2012-01-10_11h33m700s_Retailers/runtreeLog.xml` and one or several profiler logs with sequential filenames:

- `2012-01-10_11h33m700s_Retailers/profilingLog-0.xml`
- `2012-01-10_11h33m700s_Retailers/profilingLog-1.xml`

Some report executions require executing sub-queries. Sub-query execution trace files are stored under a separate directory within the main report directory. The sub-query directory contains the same logging elements as the main report, runTreeLog.xml and profilingLog-#.xml. For example, if executing the report “Retailers” requires the execution of one or more sub-queries, the trace files for those sub-queries are stored in a directory named `2012-01-10_11h33m700s_retailers/subqueries/`.

You can enable query execution tracing by following the instructions under IBM Cognos 10 Administration, Dispatchers and Services, earlier in this document.

**Query Planning Trace**

Enabling the query planning trace will write information related to the transformation of the query to the plan tree log within the `c10\logs\XQE` directory, where the `c10` represents the IBM Cognos 10 install directory. Since log files are generated for each report that is executed or validated, they adhere to the following naming convention.

```
timestamp_reportName/planningLog.xml
timestamp_reportName/planningLog_pass_###.xml
```

As an example, executing a report called “Retailers” would result in a log files named `2012-01-10_11h33m700s_Retailers/planningLog.xml`, `2012-01-10_11h33m700s_Retailers/planningLog_pass_001.xml` and so on.

If sub-queries are required, the sub-query planning trace files are stored under a subqueries directory below the main report directory, for example, in a directory named `2012-01-10_11h33m700s_Retailers/subqueries/`.

Although this trace is particularly useful when attempting to determine what decisions were made by the Dynamic Query Mode to build the execution plan, care should be taken as the resultant log
files are large and may impact overall query performance. In most cases, this trace file should be enabled only under the direction of IBM Cognos Support or IBM Cognos Development.

You can enable query planning tracing by following the instructions under IBM Cognos 10 Administration, Dispatchers and Services, earlier in this document.

Changing the Default Log Output Directory

To change the log file output to a different location, a change needs to be made to the xqe.config.xml file located within the c10\configuration directory. To do this:

1. Locate and backup the existing configuration/xqe.config.xml file.
2. Using a text editor open the original file and locate the following section:
   
   <!--logsFolder value="../../logs"/-->

3. Remove the comments and add the new physical location for the log files. For this example, the new physical location will be in the D:\logs directory on a Microsoft Windows server. When completed, the finished entry should look like the following:
   
   <logsFolder value="D:\logs"/>

4. Save the changes and close the file.
5. This change will need to be implemented on a per IBM Cognos 10 install basis and will be picked up once the IBM Cognos 10 service is stopped and started.

IBM Cognos Dynamic Query Analyzer

The IBM Cognos Dynamic Query Analyzer (DQA) is a tool that provides a graphical interface for the execution tree logs produced by Dynamic Query Mode queries. This graphical interface allows a Report Administrator to easily identify all the individual pieces of a Dynamic Query Mode query. This overview is valuable for troubleshooting of Dynamic Query Mode query performance.

You can find instructions covering installation and configuration of DQA in the IBM Cognos 10 Dynamic Query Analyzer User Guide on IBM developerWorks.

Running a Report and Viewing the Remote Logs

With the configuration of the IBM Cognos Dynamic Query Analyzer complete and the query execution trace enabled, you can run a report from within IBM Cognos Dynamic Query Analyzer and view the remote logs produced by running the report. The follow sections will provide the steps required to run a report through DQA and view the remote logs.

Running a Report within IBM Cognos Query Analyzer

1. Launch IBM Cognos Dynamic Query Analyzer.
2. By default, the Content Store view will be displayed in within the top left-hand view.
3. Within the Content Store view, locate a report and double click on it. For this example, the report name is Retailers and it is located under the GODB Essbase package.
4. Once the report finishes executing, the results will be displayed in a separate view next to the Content Store view. In this case, the report results consist of a crosstab with Retailer on the row edge, the 2005 year and its quarters on the column edge, and Quantity as the measure.
Viewing the Execution Plan (Profile) Log

1. Within the **Content Store** view, traverse the IBM Cognos 10 structure to the report executed in the previous section. For this example the report will be located under the **GODB Essbase** package as the “Retailers” report.
2. Under the “Retailers” report, a folder with a date should now be visible, expand this folder to display the **Profile 0** object.
3. **Double-click** on the **Profile 0** object. When the screen finishes loading, the profile log will be displayed in a graph within the right-hand view. Supporting the graph are the left-hand **Summary** view, the bottom left-hand **Query** and the bottom right-hand **Properties** view. This is also illustrated by the following screen capture.

**Figure 115 - DQA displaying the Graph, Summary, Query and Properties views**

Dynamic Query Analyzer in Action: A Suppression Case Study

The following section will provide a working tutorial on how the Dynamic Query Analyzer can be used to tune a report for improved performance.
In this case study a report author has created a crosstab report against the Great Outdoors sample cube provided for Oracle Essbase. The crosstab report consists of Quantity as the measure, Product as the row edge and Order Method nested under Retailer on the column edge. When creating the report, the author also selected suppression for the rows and columns from the available toolbar items. For the purpose of this case study the report is named “VisualNullSuppressionCaseStudy” under the GODB Essbase package.

**Figure 116 - IBM Cognos Report Studio Report displaying the newly created crosstab**

The analysis proceeds by opening the report’s runtree object (Profile 0) in Dynamic Query Analyzer. Within the Graph for this particular scenario, there are two items of note. The first item is the `XV5Suppress` node located near the top of the tree graph; the second is the MDX timing node, labeled `XMDXSelect` near the bottom of the tree graph. These two items are also clearly identified within the supporting left-hand Summary view.
The XMDXSelect node is the node which will display the pieces of the actual MDX query used to satisfy the report request. The scale icon beside the node is used as a visual representation of the performance of the node. The actual properties of the node are displayed by clicking on the node itself in the graph view.

The Properties view displays the node properties in a tabular format corresponding to the following excerpt from the log:

```
<XMdxSelect id="357" totalElapsedTime="59821427" ownElapsedTime="55787781"
	totalCPUPtime="46875000" ownCPUPtime="46875000" cellProperties="null" cubeName="GODB.GODB">
```

The properties reveal the time spent in the actual node and the total elapsed time. In this case, the time spent in the node itself was 56 milliseconds and the cumulative time for the node and its children was 60 milliseconds. These timings were obtained from the ownElapsedTime and totalElapsedTime elements of the XML properties.

The XV5Suppress node is evoked by the application of the Suppress\Rows and Columns on the report. By clicking on the XML view of the XV5Suppress node in the graph, the node property are displayed in a tabular format corresponding to the following XML excerpt from the profile log:
Based on the fact that there are two EdgeNum entries and nulls, divByZero, zero and overflow are all set to true, the properties confirm that a user applied zero, divide by zero, overflow and null suppression on both the rows and columns.

At this point, it would be time to ask the report author about suppression:

- Are divByZero, zero and overflow suppression actually needed?
- Would null suppression be enough?
- Is suppression actually needed on both rows and columns?

The report author replies that only null suppression is required on both columns and rows. The report author makes the change to the report and saves it as “VisualJustNullSuppressionCaseStudy” under the same package. When the runtree plan (Profile 0) for this report is viewed in the Dynamic Query Analyzer, the results show that the Suppress node in the left-hand Summary view is no longer the most costly node as the node ordering has changed. The graph supports this with a reduction of the visual performance indicator next to the suppress node. Both these points of interest are also illustrated by the following screen capture.

**Figure 118 - Graph view highlighting the timing on XV5Suppress node after running the VisualJustNullSuppressionCaseStudy report**

The XML view for the V5Suppress node now displays the following XML:

```xml
<SuppressSpec EdgeNum="1" nulls="true" divByZero="false" zero="false" overflow="false"/>
<SuppressSpec EdgeNum="0" nulls="true" divByZero="false" zero="false" overflow="false"/>
```
Only nulls are being suppressed on both the row and column edge of the crosstab. Since the requirement is to only suppress nulls, the visual null suppression can be replaced by the null suppression on the actual query.

The report author receives this request and makes the desired change. The report is saved as “QueryNullSuppressionCaseStudy” under the same package. When the runtree plan (Profile 0) for this report is viewed in the Dynamic Query Analyzer, the profile graph now appears with no V5Supression node. The supporting Summary view displays the MDXSelect node as the most costly node (in terms of performance). This is also illustrated by the following screen capture.

**Figure 119 - Graph view highlighting the timing on the XMDXSelect node after running the QueryNullSuppressionCaseStudy report**

This time, clicking on the XML view, the XMDXSelect node displays the properties corresponding to the following:

```
<XMdxSelect id="350" totalElapsedTime="21536482" ownElapsedTime="19602503" totalCPUTime="31250000" ownCPUTime="31250000" cellProperties="null" cubeName="GODB.GODB">
```

The time spent in the node itself was 19.6 milliseconds and the cumulative time for the node and its children was 21.5 milliseconds. These timings were obtained from the ownElapsedTme and totalElapsedTime elements of the XML properties. Since visual suppression has been removed from the report, the V5Suppress node is no longer present in the graph.

**Dynamic Query Analyzer in Action: A Report Comparison Case Study**

In this example, the author has written two reports to compare sales results by country. Both reports display the top 20 items, using a top-count filter. One of the reports has a detail filter to filter out all of the data that has no country information, assuming that would improve performance. The
report with a detail filter that removes empty countries is named \texttt{withCountryFilter} and the one without the filter is named \texttt{withoutCountryFilter}.

When opening a report profile in DQA, it displays a summary of the report execution. The report execution time for these two reports, displayed as Total Time in the Timing section of the Summary view, is 542 milliseconds for \texttt{withCountryFilter} and 463 milliseconds for \texttt{withoutCountryFilter}.

The report author added the detail filter on empty elements to improve performance, but it actually slowed the report down by about 80 milliseconds. Now we can find out why. By examining the Working Times and Wait Times in the Timing section of the Summary view, we can see where most of the time was taken in the execution of the report.

Let's study the timing information for \texttt{withoutCountryFilter} in more detail to see where the time is being spent (looking at the faster report first gives us something to compare to in the slower report). The following image shows the Summary and Timing sections of the Summary view for this report as well as a portion of the run tree in the Graph view.

Figure 120 - Summary timing for report withoutCountryFilter

The node with the biggest total working time and waiting time was \texttt{XQE_NS0} (46.8 milliseconds and 82.1 milliseconds). \texttt{TOPCOUNT} was also significant, with 46.8 milliseconds and 54.9 milliseconds respectively.

Compare that with the corresponding timing information for the report with the detail filter (\texttt{withCountryFilter}), as shown in the following image.
In the **Timing** section for this report profile, there are two XQE_NS nodes (compared to one in the faster report profile) and an additional FILTER node. The **TOPCOUNT** is faster (total working plus wait time of 72 milliseconds instead of 102 milliseconds). However, this saving was more than lost in the greater time consumed by the additional **XQE_NS1** and **FILTER** nodes (a total of 236 milliseconds) compared to the time recorded for the single XQE_NS0 node in the faster report (129 milliseconds).

So what is going on in those XQE_NS nodes? We can look in the graph to see the tree but that doesn't tell us much about the query that was generated for them.

If we continue to analyze **withCountryFilter**, we can open the **Query** view (it opens by default but if you have closed in you can find it with **Window > Show View** from the main menu). Since this report is based on a Dimensionally Modeled Relational package, the **Query** view displays the MDX that was generated to run the report when the graph is selected, as shown in the following image.
In the query text, **XQE_NS1** is the entry for the detail filter to remove data with no country and **XQE_NS0** is the entry for the **TOPCOUNT**. This confirms our conclusion that the detail filter cost more time than it saved in executing this report.

If we then want to look in the graph at the nodes associated with the query we can select the text in the **Query** view and then click on the **Link MDX to Graph** button on the **Query View** toolbar. That will then take us into the graph and show the node in question.
Submitting a Dynamic Query Mode Test Case to IBM Cognos Support

In addition to what IBM Cognos support requests for query related support incidents, the following items should be submitted for a Dynamic Query Mode query diagnosis:

1. A detailed description of the query problem, along with the desired query output or expected query behaviour.
2. Data source information such as:
   - Data source type (Oracle Essbase, SAP BW, TM1)
   - Data source version
   - Connection string
   - Connectivity client version
4. <code>config/XQE</code> properties file for the specific data source type.
5. <code>config/qfs_config.xml</code> file.
6. IBM Cognos 10 Framework Manager model.
7. Query execution trace.
8. Query planning trace
Downloadable resources

<table>
<thead>
<tr>
<th>Description</th>
<th>Name</th>
<th>Size</th>
</tr>
</thead>
<tbody>
<tr>
<td>IBM Cognos BI 10.1 PDF version of this document</td>
<td>IBM_Cognos_10_1_Dynamic_Query_Cookbook.zip</td>
<td>1768KB</td>
</tr>
</tbody>
</table>

© Copyright IBM Corporation 2010, 2013

Trademarks
(www.ibm.com/developerworks/ibm/trademarks/)