IBM Cognos 8 Business Intelligence
Transformer

Version 8.4.1

User Guide

Information Management software

Product Information

This document applies to IBM Cognos 8 Version 8.4.1 and may also apply to subsequent releases. To check for newer versions of this document, visit the IBM Cognos Information Centers (http://publib.boulder.ibm.com/infocenter/cogic/v1r0m0/index.jsp).

Copyright

Table of Contents

Introduction

This document is intended for use with IBM Cognos Transformer, the OLAP modeling component delivered with IBM® Cognos® 8 Business Intelligence.

This *User Guide* describes PowerCube modeling procedures and concepts, product functionality, and related terminology. There is reference information that supplements the task- and process-oriented topics, as well as IBM Cognos 8-specific troubleshooting tips and detailed help for the more commonly encountered error messages.

You can use this document to help you model and build PowerCubes with the Transformer (Windows) interface, or to perform production-related tasks from the Windows, UNIX, or Linux command line.

For information about creating automation scripts using Model Definition Language (MDL), see the Transformer *Developer Guide*.

Audience

This document is for new Transformer users and IBM® Cognos® Series 7 cube modelers who are seeking guidance as they migrate their PowerCubes and related applications to the IBM Cognos 8 environment. Advanced database administration (DBA) or data modeling skills are not required. Business-relevant examples, samples, and code examples are supplied in context.

Related Documentation

Our documentation includes user guides, getting started guides, new features guides, readmes, and other materials to meet the needs of our varied audience. The following documents contain related information and may be referred to in this document.

Note: For online users of this document, a Web page such as **The page cannot be found** may appear when clicking individual links in the following table. Documents are made available for your particular installation and translation configuration. If a link is unavailable, you can access the document on the IBM Cognos Resource Center (http://www.ibm.com/software/data/support/cognos_crc.html).

Documents	Description
IBM Cognos Connection User Guide	Using IBM Cognos Connection to publish, find, manage, organize, and view IBM Cognos content, such as scorecards, reports, analyses, and agents
IBM Cognos 8 Architecture and Deployment Guide	Understanding the IBM Cognos 8 architecture, developing installation strategies, including security considerations, and optimizing performance

Documents	Description
IBM Cognos 8 Getting Started	Teaching new users how to use IBM Cognos 8
IBM Cognos 8 Installation and Configuration Guide	Installing, upgrading, configuring, and testing IBM Cognos 8, changing application servers, and setting up samples
IBM Cognos Transformer Developer Guide	Modeling and building PowerCubes using Model Definition Language (MDL) scripts
Framework Manager User Guide	Creating and publishing models using Framework Manager
Framework Manager Developer Guide	Creating and publishing models using the Framework Manager API
IBM Cognos 8 Administration and Security Guide	Managing servers, security, reports, and portal services; and setting up the samples, customizing the user interface and troubleshooting
Analysis Studio User Guide	Exploring, analyzing, and comparing dimensional data

Finding Information

Product documentation is available in online help from the **Help** menu or button in IBM Cognos products.

To find the most current product documentation, including all localized documentation and knowledge base materials, access the IBM Cognos Resource Center (http://www.ibm.com/software/data/support/cognos_crc.html).

You can also read PDF versions of the product readme files and installation guides directly from IBM Cognos product CDs.

Getting Help

For more information about using this product or for technical assistance, visit the IBM Cognos Resource Center (http://www.ibm.com/software/data/support/cognos_crc.html). This site provides information on support, professional services, and education.

Printing Copyright Material

You can print selected pages, a section, or the whole book. You are granted a non-exclusive, non-transferable license to use, copy, and reproduce the copyright materials, in printed or electronic format, solely for the purpose of operating, maintaining, and providing internal training on IBM Cognos software.

Chapter 1: What's New?

This chapter contains a list of new, changed, and deprecated features for this release. It also contains a cumulative list of similar information for previous releases. Knowing this information will help you plan your upgrade and application deployment strategies and the training requirements for your users.

For information about upgrading, see the *Installation and Configuration Guide* for your product.

For an overview of new features for this release, see the *New Features Guide*.

For changes to previous versions, see:

- New Features in Version 8.4
- New Features in Version 8.3

To review an up-to-date list of environments supported by IBM Cognos products, such as operating systems, patches, browsers, Web servers, directory servers, database servers, and application servers, visit the IBM Cognos Resource Center Web site (http://www.ibm.com/software/data/support/cognos_crc.html).

New Features in Version 8.4

OLE Automation Support

Support for OLE automation has been reintroduced in Transformer version 8.4. This latest OLE automation includes support for new security features and other Transformer features introduced in version 8.4. For more information about Transformer OLE automation, see the *Transformer Automation Guide*.

TM1 Support

Transformer now supports TM1 cubes as a data source in Transformer.

For more information about using OLAP data sources such as TM1, see "Data Source Types" (p. 38).

IBM InfoSphere Warehouse Cubing Services Support

Transformer now supports the cubing services technology of IBM InfoSphere Warehouse as a data source.

For more information about using OLAP data sources, see "Data Source Types" (p. 38).

Excel 2007 Support

Transformer now supports Microsoft Excel 2007 .xlsx files as a data source.

For more information, see "Data Source Types" (p. 38).

Welcome Page

To align IBM Cognos Transformer with the IBM Cognos 8 suite of products, Transformer now shows a welcome page when the product opens.

From the welcome page, you can create a new model, open and edit an existing model, and view a list of recently used models. For more information about the welcome page, see "The Interface of Transformer Version 8.x" (p. 333).

SAP BW Data Source Support

You can now use IBM Cognos Transformer to import both dimensional and fact data from a specifically constructed SAP BW queries source.

For more information, see "Guidelines for Working with SAP BW Data for Use in Transformer" (p. 361).

Copy and Activate New Versions of PowerCubes

To make new versions of published PowerCubes available to users, you can copy new versions to the location where the published version exists, and then activate the newer version.

For more information, see "Copy and Activate a Newer Version of a Published PowerCube" (p. 210).

Set Suppression Options for IBM Cognos 8 Packages

You can set suppression options for packages published with a PowerCube. These options determine whether IBM Cognos 8 studio users can choose to hide empty rows only, empty columns only, or both empty rows and columns. Options also determine the types of empty values that can be suppressed, such as zero or missing values. Types of empty values that users can choose to suppress depend on the IBM Cognos 8 studio.

For more information, see "Publish a PowerCube" (p. 186).

Multiple Hierarchy Support in IBM Cognos 8 Studios

IBM Cognos 8 studios now support multiple hierarchies from the same dimension in a report. Hence, it is now possible to author reports that reference more than one hierarchy within a single dimension. For more information, see your studio documentation.

Integration with Business Viewpoint Studio

Transformer uses the Business Viewpoint Client to connect to the Business Viewpoint master dimension management system. The integration permits a Transformer user to subscribe to a master dimension from the system and to send Transformer updates to the system, and receive updates from the system. You can nominate dimensions from model files that are saved both as .pyj and .mdl. Complex models built using MDL scripting work with Business Viewpoint Client.

Changed Features in Version 8.4

Automatically Synchronize Columns with IBM Cognos 8 Data Sources

As the modeler, you must ensure that the columns in your model reflect the current state of your data sources. If, after a data update, mismatches occur between IBM Cognos 8 data source items and columns in the model, you can resynchronize them automatically from a list of possibilities provided by Transformer.

For more information, see "Synchronize Columns with Your Data Source" (p. 62).

Composite Information Server is Replaced By IBM Cognos 8 Virtual View Manager

Composite Information Server was available with earlier releases of IBM Cognos 8. In the current release, Composite Information Server is replaced by IBM Cognos 8 Virtual View Manager, which is an IBM proprietary product that is based on a new version of Composite Information Server. In this release, the default repository is changed, from Microsoft SQL Server to IBM Informix. If you have Composite data sources defined in IBM Cognos Connection, you must migrate the existing repository to the new default repository. For more information about migrating the repository, see the IBM Cognos 8 Virtual View Manager *User Guide*. For more information about data source connections, see the *Administration and Security Guide*.

Deprecated Features in Version 8.4

There are no deprecated features for Transformer version 8.4.

New Features in Version 8.3

Transformer version 8.3 is fully integrated with the IBM Cognos 8 security, data sources, and publishing capability.

Listed below are new features since the last release. Links to directly-related topics are included.

Transformer Version 8.3 on All IBM® Cognos® 8 Business Intelligence Version 8.3 Supported Platforms

For the first time, Transformer runs on all supported IBM Cognos 8 platforms, including Linux, HP/UX Itanium, and Windows Vista. For information about installing Transformer version 8.3 on these platforms, see the IBM Cognos 8 *Installation and Configuration Guide*.

For an up-to-date list of supported environments, visit the IBM Cognos Resource Center Web site (http://www.ibm.com/software/data/support/cognos_crc.html).

Transformer Version 8.3 Installation Download

Transformer can now be made available more easily for business specialists who want to design models and build PowerCubes for their own use. For example, IT departments can provide business specialists or Transformer modelers with a Web-based, downloadable installation program from a corporate or secured portal, allowing for easy distribution of the installation files.

For more information, see "Install IBM Cognos Transformer" in the IBM Cognos 8 *Installation and Configuration Guide*.

Integration with IBM Cognos 8 Business Intelligence Security

Transformer version 8.3 supports simultaneous user authentication and logon using the full range of supported IBM Cognos 8 security providers.

In previous versions of Transformer, security was restricted to the use of user classes, and each custom view could be configured with only one user class. In Transformer version 8.3, you use IBM Cognos 8 users, roles, or groups as security objects, and assign multiple security objects to each custom view.

Secured cubes created in Transformer version 8.3 are intended for the IBM Cognos 8 Business Intelligence Web studios and are not backward compatible with IBM Cognos Series 7 PowerPlay products.

Unsecured and password-protected PowerCubes built in Transformer version 8.3 can be accessed in IBM Cognos 8 Mobile Analysis version 8.3 for local (disconnected) use.

For information about Transformer version 8.3 secured cubes, see "Adding Security" (p. 173).

Upgrade User Class Views to Transformer Version 8.3 Custom Views

When you upgrade IBM Cognos Series 7 secured PowerCube models, you have the ability to move from IBM Cognos Series 7 Access Manager security to the authentication provider configured in your IBM Cognos 8 Business Intelligence version 8.3 installation.

You can choose to continue to use the IBM Cognos Series 7 namespace, or you can add a security object from a different authentication provider to your upgraded views. When you add a security object from a different authentication provider, you do not lose the dimensional filtering, such as apexing or cloaking, included in each view.

For information about Transformer version 8.3 secured cubes, see "Adding Security" (p. 173).

Support for Multiple Namespaces

During the transition from an IBM Cognos Series 7 namespace to an alternate security provider, you can use the PowerCube property **All applicable namespaces** to associate all applicable namespaces during migration testing. When you associate all the applicable namespaces to the cube, you can ensure that the group, role, or user dimensional filtering is consistent with that which had been applied for the IBM Cognos Series 7 user class. This option is supported only for migration testing, and cannot be used to deploy cubes in production environments.

The **All applicable namespaces** option is found on the **PowerCube** property sheet (**Data Source** tab) and in the IBM Cognos 8 **New Data Source** wizard.

Integration with IBM Cognos 8 Business Intelligence Version 8.3 Data Sources

You can import metadata from IBM Cognos 8 packages and reports, including the associated filters and prompts, for use in defining data sources in Transformer version 8.3.

The IBM Cognos 8 package or report is a container data source; you can define multiple data source queries associated with each package or report that you create as a data source in Transformer version 8.3. You can create data source queries using IBM Cognos 8 reports created in Query Studio or Report Studio using relational or DMR packages. You can import query items to create your structural or transactional queries.

You cannot create data source queries using IBM Cognos 8 reports that contain OLAP data sources.

For information about using IBM Cognos 8 packages and reports as Transformer data sources, see "Create a Model" (p. 47).

Using OLAP Packages

Using the Transformer version 8.3 dimension map, you can import dimensions from packages based on OLAP data sources. This can be useful in creating conformed dimensions and, to some extent, to reusing portions of the published metadata from the source dimension.

You cannot import time dimensions from OLAP packages if you want to take advantage of Transformer's relative time functionality. You must import fact data using other sources, such as flat file exports or relational packages and reports.

For information about using IBM Cognos 8 packages and reports as Transformer version 8.3 data sources, see "Create a Model" (p. 47).

Unicode Support

Transformer version 8.3 includes limited support for Unicode. PowerCubes can be built with only one locale, however you can now leverage data sources with multiple locales:

- Transformer supports Unicode data sources, such as Framework Manager packages, IBM Cognos 8 reports, and exported .csv files.
- Transformer can be installed and build PowerCubes on an operating system configured to use UTF-8, a Unicode encoding.

For more information, see "Managing Languages and Locales" (p. 185).

Series 7 IQD Bridge

IBM Cognos Transformer uses the IBM Cognos 8 query engine to support the features integrated with IBM Cognos 8. The Series 7 IQD Bridge component enables Transformer version 8.3 to continue supporting IBM Cognos Series 7 .iqd files, whether the files were authored in IBM Cognos Impromptu, or in Framework Manager as externalized queries. To use an .iqd data source with Transformer version 8.3, you must install the Series 7 IQD Bridge on each Transformer computer that requires access to this data source type.

The Series 7 IQD Bridge is an optional component during installation, but is not available on new platforms such as Linux or HPUX Itanium.

If you have Framework Manager externalized queries as a data source, we strongly recommend that you publish the query subjects in a package, and the query items be imported in Transformer version 8.3 using the package as a data source. Alternatively, you can create a report based on the package and import the report as a data source. This method provides functionality that was unavailable in externalized queries in previous releases, such as support for multi-select statements, local processing, and multi-database connections. For more information about using IBM Cognos 8 packages as a data source, see "Create a Model" (p. 47).

For more information about the Series 7 IQD Bridge, see the IBM Cognos 8 *Installation and Configuration Guide*.

Composite Source Packages

Transformer version 8.3 supports Composite Server-sourced published packages.

Publishing Integration with IBM Cognos Connection

When your IBM Cognos 8 environment is properly configured and the Transformer modeler has the required permissions, you can publish any cube in the PowerCubes list directly to IBM Cognos Connection. You must specify the Windows or UNIX/Linux location of the cube so that the IBM Cognos 8 Business Intelligence server can access it. You can publish cubes to IBM Cognos Connection using the Windows user interface or from the command line on any supported platform.

Publishing to IBM Cognos Connection directly from Transformer version 8.3 allows you to publish one cube per package. However, with IBM Cognos 8 version 8.3, you can publish packages into folders; with this functionality, you can better manage published packages in IBM Cognos Connection.

In previous IBM Cognos 8 releases, you published cubes to IBM Cognos Connection using Framework Manager. With Transformer version 8.3, you can still use Framework Manager to publish multiple cubes per package or include your PowerCubes in a package with other data sources. For more information about publishing cubes from Framework Manager, see the Framework Manager *User Guide*.

For more information about publishing Transformer version 8.3 cubes to IBM Cognos Connection, see "Publish a PowerCube" (p. 186).

Cube Update Utility

With IBM Cognos 8 version 8.3, you can use a command line utility to update PowerCubes in production environments. This utility uses the IBM Cognos 8 SDK, and provides cube updates with little to no impact on end users.

For more information, see "Update Published PowerCubes and PowerCube Connections" (p. 210).

Changed Features in Version 8.3

Listed below are changes to Transformer functionality since the last release. Links to directly-related topics are included.

Reordering Data Sources

In previous versions of Transformer, we recommended that, to improve cube build performance, fact queries appear as the last data sources in the Data Sources list. To achieve this result, you could reorder the data sources in the **Data Sources** list.

In Transformer version 8.3, the order of the data sources does not impact cube build performance.

Enabling Multi-processing

In previous versions of Transformer, the data source properties property sheet included an option for enabling multi-processing. In Transformer version 8.3, multi-processing is no longer a user option. Instead, Transformer version 8.3 detects when multi-processing is appropriate, and enables the setting for applicable data sources.

Note: For Transformer version 8.3 to enable multi-processing automatically, the Series 7 IQD Bridge must be installed.

When IBM Cognos Series 7 models are imported in Transformer version 8.3, Transformer maintains the multi-processing setting where appropriate.

IBM Cognos Configuration

Transformer version 8.3 uses IBM Cognos Configuration to configure the server environment. This enables complete integration with all IBM Cognos 8 Business Intelligence products.

For more information, see the IBM Cognos 8 *Installation and Configuration Guide*.

IBM Cognos 8 Installation Location

In previous releases of Transformer, the installation location was a cer*n* directory, where *n* represented the Transformer version 7.x rendition number. The Transformer version 8.3 installation location is the same c8 directory used by all other IBM Cognos 8 products. This allows Transformer version 8.3 to be installed on the same computer as previous versions of Transformer.

The Transformer version 8.3 executable name is now cogtr.exe on Windows, and cogtr on UNIX/Linux. The Transformer preferences file name (previously the .ini file) is now cogtr.xml. The cogtr.xml file is installed in the *installation_location*/configuration directory. Log files for Transformer version 8.3 are maintained in the My Documents/Transformer/Logs directory on Windows, in the Documents/Transformer/Logs directory on Windows Vista, and in the *installation_location*/logs directory on UNIX/Linux.

Note: When Transformer version 8.3 is installed on Windows Vista, if you do not run Transformer in elevated mode and you make changes to the cogtr.xml file, the updated file is saved by default to a **Virtual Store** directory and not to the *installation_location*/configuration directory. To run Transformer in elevated mode on Windows Vista, right-click Transformer (for example, in the **Start** menu) and then click **Run as Administrator**.

For more information, see the IBM Cognos 8 *Installation and Configuration Guide*.

One Version of Transformer

Previous releases of Transformer included three different versions:

- Personal Transformer
- Transformer for UNIX Client (Windows version)
- Transformer Server (UNIX)

IBM Cognos Transformer version 8.3 has only one version of Transformer, which you can install on any IBM Cognos 8 supported platform. Modeling continues to be carried out on the Windows client.

Transformer Version 8.3 Online Help

In previous versions of Transformer, online help was available in WinHelp format, with each context-sensitive help topic appearing in a separate, small dialog box.

In Transformer 8.3, help, including context-sensitive help, is available in HTML format. When you click the **Help** button in a dialog box, an HTML page opens. The HTML page contains all the topics associated with the dialog box. For information about the main Transformer window, see "Reference" (p. 333).

Deprecated Features in Version 8.3

A deprecated feature is one that is being replaced by a newer version or a better implementation. We intend to discontinue the use of the feature and provide recommendations for alternative practices to adapt to this change over multiple releases.

Listed below are deprecated features, including links to related topics or further details.

.pyi Files

In Transformer version 8.3, models saved in binary format now use the .pyj file extension. In Transformer version 7.x, models saved in binary format used the .pyi file extension.

Before upgrading Transformer version 7.x models to Transformer version 8.3, you must save your .pyi models in the .mdl (text) file format. We strongly recommend that you keep a .mdl file backup of all IBM Cognos Series 7 models that you upgrade to Transformer version 8.3.

Data Source Types

Support for the following data source types has been discontinued:

- dbase table
- paradox table
- Lotus 1-2-3 crosstab
- Lotus 1-2-3 database
- FoxPro table
- Clipper table
- Architect models

Client/Server Functionality

Automatic synchronization between Transformer Windows and UNIX server operations, using the client/server functionality, is not included in Transformer version 8.3. As a result, the Server menu was removed in the Windows user interface.

To deploy models and cubes for use on UNIX or Linux, we recommend porting the models and cubes to the UNIX or Linux computers.

Customized Menu Options

Previous versions of Transformer allowed users to customize the menu options. This functionality is not included in Transformer version 8.3.

Compressed PowerCubes

Previous versions of Transformer allowed users to compress PowerCubes. This functionality is not included in Transformer version 8.3.

IBM Cognos Transformer Expression Editor Functions

Transformer version 8.3 continues to use the expression editor included with previous Transformer releases. However, the following expressions are no longer supported:

- Pack
- Spread
- Substitute
- Phdatetodate

Chapter 2: Planning Your Model

IBM Cognos Transformer is a data modeling tool designed for use with IBM Cognos 8 version 8.3 and subsequent releases.

You use this component to create a model, a business presentation of the information in one or more data sources. After you choose a supported product locale (language), add dimensional metadata, specify the measures (performance indicators), and apply custom views, you can create PowerCubes based on this model. You can deploy these cubes to support OLAP reporting and analysis around the globe.

This section provides a high-level overview of the modeling process and recommendations for planning a model design to meet the OLAP needs of your users, as well as information about how to upgrade an IBM Cognos Series 7 Transformer model.

The documented workflow follows a logical sequence, beginning with analyzing your requirements and building a prototype model. If you have already completed this planning stage, you can proceed to the sections of this document that deal with data sources (Chapter 3), dimensions (Chapter 4), and measures (Chapter 5).

Dimensional Modeling Workflow

IBM Cognos Transformer is a proven and relatively simple tool for modeling dimensional hierarchies and levels for PowerCubes.

After you relate the dimensions to your business performance indicators, you can create powerful, secure cubes to be used for reporting and drill-through analysis in the IBM Cognos 8 Web studios.

We recommend that you use the following workflow:

- ❑ Carefully analyze your users' OLAP reporting requirements.
- ❑ If you have not already done so, build a prototype model.
- ❑ Choose your transactional and structural data sources and import the facts (measures) and metadata (dimensions).
- ❑ Map your metadata into dimensions, and your facts into measures.
- ❑ Verify the model and resolve any ambiguities.
- ❑ Organize the data in your model into customized dimension views or cube groups.
- ❑ Apply security and create custom views to control access to sensitive information.
- ❑ Create and publish PowerCubes to IBM Cognos Connection.
- ❑ Manage and maintain your models, cubes, and reports for optimal effectiveness.

Troubleshooting tips are provided in this document and in the IBM Cognos 8 *Administration and Security Guide*. This document also provides an overview of the functions supported by Transformer, and how they may be used to create calculated expressions. For more information, see "Transformer Functions" (p. 126).

For information about scripting in Model Definition Language (MDL), see the Transformer *Developer Guide*.

Analyzing Your Requirements and Source Data

To ensure that you develop an effective business intelligence model, we recommend that you begin by carefully analyzing your users, the OLAP reports they require, and your source data.

Use the following questions to analyze your users' OLAP reporting needs:

- What reports do users currently use? Which reports do they use most frequently? Which reports do they use only rarely?
- Does each group require different reports? Are there some reports that are required by all user groups?
- Do users need higher-level (summary) reports, detailed drill-through reports, or both?
- How frequently are the measures in the report updated? How frequently do the reports themselves change? Does the frequency vary from group to group?
- How often are reports required? Can you trade off frequency to ensure accuracy? For example, if your users ask for monthly reports and the data source is refreshed weekly, the data will always be current. However, if your users want daily reports, the data will only be up to date on the first day of the weekly cycle.

Analyze your source data, using questions such as the following:

- Does the data come from one source or many? What format is it in: flat files, spreadsheets, or databases? Does it need to be converted to a supported data source type before it is imported?
- Can you optimize existing queries by building new Transformer queries using the metadata modeled in IBM Cognos 8 packages or reports?
- How many records are there? By how much do you expect the volume of data to increase?
- How much of the data is static and how much changes gradually over time? Can you create different data sources for static and non-static data to support incremental updates (an option that shortens cube creation time by appending new data to a cube instead of recreating it)?
- How much data preparation is required?

  We recommend that you ensure that the source values that feed the categories are unique and, if feasible, that you aggregate or otherwise preprocess your data before importing it. For more information, see "Recommendation - Presort, Clean, or Consolidate Your Data" (p. 27).
- Are linked measures from different data sources updated at the same time?
- Must you create additional data sources to accurately model your organization?

When you have answered these questions, you are ready to begin preparing your source data for import and designing your prototype.

Recommendation - Presort, Clean, or Consolidate Your Data

By preprocessing your data, you can maximize reporting flexibility and performance.

Preprocess data to achieve the following benefits:

- Presorted records are processed more quickly in Transformer.
- When you streamline your source data to contain only the information needed for the model, read times are faster in Transformer.
- You can use Transformer to presummarize the data when your users do not require access to all the details in the source.

  For example, if your organization processes 50,000 transactions daily, and you create the cube weekly, you can summarize the transactions at the weekly level before Transformer begins processing. This will greatly speed up cube creation.
- Consolidation, combining records with identical non-measure values, reduces the size of the cube and improves performance in your reporting application.

  Consolidation is enabled by default in Transformer. Evaluate your data to see if it can be further consolidated by using the **Duplicates rollup** or **Regular rollup** features of Transformer.

  For consolidation purposes, non-measure values are considered identical if they meet any of the following criteria for the particular rollup:

  - The source data contains transactions with identical non-measure values.

    For example, two sales of the same product are made to the same customer on the same day, but the colors differ. If colors are omitted from a dimension view using the **Suppress** or **Summarize** command on the **Diagram** menu, the sales records will have identical non-measure values.
  - Records become identical when a dimension is omitted from the cube.

    For example, two sales of the same product are made at different stores on the same day. If the Stores dimension is removed from the model, these sales records will have identical non-measure values.
  - Records become identical because of the **Degree of detail** setting on the **Time** tab of the **Column** property sheet.

    For example, if the **Degree of detail** is set to Month for a column associated with a time dimension that includes week and day values, Transformer ignores the week and day values in the source transactions when consolidating records.
- For queries based on relational packages, enabling the **Auto summarize** feature on the **General** tab of the **Data Source** property sheet also helps reduce the number of rows that Transformer retrieves from the source data, further improving cube build performance.

Separate Your Structural and Transactional Data

Processing time improves when Transformer can query your structural and transactional information separately. We recommend that you identify which data sources contain purely structural information, which contain transactional information (measure values or facts), and which contain a combination of the two.

When processing queries to create a PowerCube, Transformer orders the queries, first reading the structural queries and then reading the transactional queries.

Ideally, you should define each dimension or drill-down path with a separate structural data source, and then add one or more transactional data sources to provide the measures for those dimensions. This restructuring exercise helps to partially normalize your data, speeding up both the category generation and cube creation stages.

The best approach is to have unique levels near the bottom of the dimensions, and to have the transactional queries link to the dimensions using those levels. This is basically the star schema or snowflake method of creating dimensions in a relational database. Processing is faster because each transaction record has fewer business keys to process in identifying the category that the measure values are associated with, at the bottom of the dimension.

We recommend defining any transactional data sources that change frequently so that they contain a small, concise record set, with the minimum amount of information needed to update the data in your PowerCubes. Whenever possible, save your model with generated category structures, to eliminate the redundant processing required to continually rebuild them. Similarly, if your model contains long descriptions, we recommend that you generate cubes from a model that is already populated with the categories associated with those descriptions.

For more information, see "Control When the Source Data Is Read" (p. 71).

Additional Data Modeling Tips

To enhance your model design, consider building the following steps into your process:

- Analyze the data flow from the point at which your data is generated until the data is input into Transformer. Determine if the data can be streamlined or rationalized at any point, perhaps by creating a data warehouse, a series of data marts, or a data-extract process to reorganize it.

- Resolve uniqueness issues and data dictionary terms before you merge two sets of data into one model. Ensure data integrity by checking your column joins; outer joins or table aliases may be required. Remember that Transformer is not a relational database tool, and cannot perform joins between the columns of different data sources. If you need to set up database joins, use a modeling tool such as Framework Manager to create the joins, and then publish the Framework Manager package for use in Transformer.

- Wherever possible, build flexibility into your plan. Use a different source file for each aspect of your business, and organize the data sources in your model so that each data source supplies the data for a different dimension. That way, you can add more information into your cube as your business evolves, even if the data comes from different software applications, platforms, departments, or locations.

- Improve performance by continually striving to reduce the Transformer processing load.

Building a Prototype

To field-test the accuracy of your analysis, build an initial model or prototype that reflects the needs of the key decision makers in your company.

Base your prototype on an existing set of frequently used, stable OLAP reports, and use the following checklist:

❏ Identify Measures

Measures are the numbers you use to gauge your organization's performance. You should choose the critical success factors in your business as your measures. Examples of typical measures include sales revenues, profit margins, and response times.

If you have multiple data sources, you must relate the dimensions and levels of your model to the data source that contains the columns to be used for each measure.

Your model is more effective if your measures are applicable to more than one dimension. For example, if your dimensions are products, locations, and customers, your measures should bridge these dimensions.

❏ Specify a Time Dimension

To ensure that your users can make period-to-period comparisons and visualize trends over time, choose a time dimension that reflects and synchronizes accounting periods and reporting schedules.

In most cases, your requirements are met by models based on the calendar or fiscal year. Month, Quarter, and Year categories can be supplemented by relative time categories automatically generated by Transformer, such as **YTD Growth**, the percent-growth year-over-year.

If your organization uses particular time periods, such as lunar weeks and months, or three 8-hour shifts per day, Transformer supports the definition of custom time dimensions. Even if your query objects originate in Framework Manager, we recommend that you import the necessary time-related items into Transformer, and then define your time dimensions there.

❏ Select the Data to be Modeled

You begin by identifying the data sources that contain the data for the model you want to create.

Suppose that information about your customers is stored in a Customers table and information about your products is stored in a Products table. Related tables called Customer_Details and Product_Details provide additional information about customers and products. Order information is stored in two tables called Orders and Order_Details.

In keeping with good design practice, you decide to set up the Customers, Customer_Details, Product, and Product_Details tables as structural data sources, to provide the information that Transformer uses to build the Customers and Products dimensions in your model.

The information about transactions is stored in the Orders and Order_Details tables. For efficiency, you decide to combine the information in these tables into a single data source called Order_Info.

The Order_Info data source contains the following information, all of which you use to associate sales with particular customers and products:

- The order dates generate categories for the time dimension.
- Data about customers and sales representatives generates the header information.
- The product, order quantity, and sales amount for each line item in an order provide the sales measures.
- The cost of the order and discounts applied to it provide supplementary fact data.

Example - Your Prototype Sales Model, on Paper

You can create an initial dimension map on paper, to make sure you have identified all of the dimensions, levels, and categories needed in your PowerCube. The measures to be associated with this dimensional hierarchy are Sales, Order Qty, Cost, and Discount.

You map the dimensions of your prototype as follows:

Order Date	Products	Locations
Year	Product Group	Region
Quarter	Product Class	Office
Month	Product Name	Sales Rep
	Product No	

Refining Your Model

Based on your paper prototype, you create the **Dimension Map** for your new model in Transformer. You begin with one data source. You can enhance the business value of your model later, by adding more sources or manipulating the data derived from the existing data sources.

Suppose you are initially lacking information about the staffing levels in each branch. You can either add another data source to provide this information or use the **Category Count** feature of Transformer to provide this detail. The resulting cube and OLAP reports can then deliver value-added information about the average sales per employee.

Models can contain any combination of the following:

- regular measures, or the numeric fact data found in a transaction file
- calculated measures, or numeric data calculated from other measures, mathematical operators, and numeric constants
- category counts, or the number of categories in a unique level for which the measure values are not zero or missing
- calculated categories, whereby calculated expressions apply directly to any measure

- calculated columns, whereby new data is based on values calculated from other columns, functions, constants, and calculated columns

Use the following checklist to help refine your model:

❑ Add special categories to enable quicker data access.

Group your data based on attributes that may be contributing to the success of your enterprise, such as product color or customer income.

❑ Add drill-down paths to provide more detail.

A dimension normally consists of a single drill-down path with one or more drill-down levels, representing the hierarchical organization of the information. However, you can further subdivide your dimensions, so your report users can analyze their data at different levels of detail.

There are no restrictions on the number of levels and drill-down paths that you use in a dimension. However, all alternate drill-down paths must converge at a common unique level and, for performance reasons, it is best to keep a 1:10 ratio or less between the categories in each level.

For information on drill through using categories from alternate drill paths, see the IBM Cognos 8 *Administration and Security Guide*.

❑ Allocate measures to other levels or dimensions.

If your model uses multiple data sources, consider allocating measures to levels or dimensions with which they are not normally associated. Allocation can provide you with new insights into your data. For example, you can associate resource-related data to financial data.

You can allocate measures over entire dimensions, over levels within an individual dimension, or over categories within levels. When allocating measures, use measures that come directly from your source data rather than calculated measures, and avoid overloading your model with superfluous detail.

❑ Consider combining information from another functional area, such as materials and resource planning or performance quality, with the finance or customer profitability data already in your business model.

Begin by listing the data columns and determining if there are any gaps, particularly in the area of cost of materials, or indicators of quality.

Next, map the new dimensions, checking that the time periods are consistent with each other and with your financial statements. Ensure that revenue and expense values map to those in the financial statements.

Finally, verify the relationships that exist between the various measures. If these are not one-to-one relationships, confirm how each relates to your common dimensions.

Example - Adding Customer Service Data to Refine Your Model

Suppose your initial model includes the following dimensional hierarchy, as well as values for Inventory Status and Turnover Ranges.

You have data for an extensive list of measures: Sales, Order Qty, Material Cost per Unit, Discount, Percent Gross Margin, Carrying Cost per Unit, Percent Material Cost per Sale, Percent Carrying Cost per Sale, Sales per Customer, Percent Profit per Segment, and Inventory Turnover.

You decide you want to monitor customer service, so you expand your model to include indicators of service quality. The new dimensions and categories might be encoded Reasons for Dissatisfaction or Causes of Poor Quality Service.

You must ensure that your source data provides the required measures, such as the number of complaints, returns, and claims, or the dollar value of returns and claims.

You can complete your model by incorporating response times, labor costs, time lost to service claims, rework hours, scrap costs, or any other factor that significantly affects service quality.

Recommendation - Diagnose and Resolve Any Design Problems

You can use any or all of the following tools and techniques to diagnose and resolve problems in your model design.

Show Scope

To see how your measures and levels are associated with their corresponding data sources by allocation, direct association or indirect association, use the **Show Scope** command on the **Edit** menu.

Show Count

To verify that you have maintained a 1:10 ratio or less between the categories in each level, use the **Show Counts** command on the **Edit** menu. Lower ratios allow for efficient partitioning and faster cube creation times in Transformer, as well as easier data exploration in your reporting component.

Show Reference

To confirm the origin of every data source column associated with your **Dimension Map** and see how each is used, use the **Show Reference** command on the **Tools** menu.

Generate Categories

To confirm how the categories in a specific data source relate to your model, use the **Generate Categories** command on the **Run** menu, with the selected data source. To prevent the generation of categories in specific levels or entire dimensions, select the **Prohibit automatic creation of new categories** check box on the **General** tab of the **Level** or **Dimension** property sheets.

Create Selected Cubes

During the prototyping stage, you may want to create only certain cubes. You can enable or disable cube creation in one of following ways:

- Change the **Cube creation** option on the **Processing** tab of the **PowerCube** or **Cube Group** property sheet.

- Use the **Create Selected PowerCube** command on the **Run** menu.

- Use the Model Definition Language (MDL) function `CreateFromCubes`. For more information, see the Transformer *Developer Guide*.

Check Cube Build Status

When you build a cube in Transformer version 8.x, you can check the status of the cube build at any time without opening it by using the **PowerCube Status** command on the **Tools** menu. You can check the status of all the cubes that are defined for a model at the same time. If your model has more than one cube, you can apply a filter to monitor the status of cubes enabled for creation, disabled cubes, or both.

You can also filter the cube build status settings by selectively requesting one of the following:

- **Any status**, to list all cubes associated with the model, regardless of their status.
- **Errors**, to list cubes that were not created because they are not valid, or failed.
- **Warnings**, to list all cubes for which warnings were detected during a previous create.
- **Successful**, to list all cubes created without errors or warnings, having a status of OK.

Consult the Error Message and Troubleshooting Help

In addition to the troubleshooting topics in the *User Guide*, help is available from **Help** buttons in some error messages to help you resolve any model design problems.

Review the Resulting Reports With Your Users

After you generate a few reports from your prototype, we recommend that you ask for feedback from representative users by posing open-ended questions. If you are the IT specialist, involve an experienced business analyst in the process.

Together, try to develop and maintain a list of follow-up questions, such as the following:

- Does each dimension level generate valid data, with measures that are properly associated or coordinated, for every data source?

  Try to spot measures that do not roll up as expected, or that are not additive in every dimension.

- Are ranges or qualitative values coded realistically? Are the values for key performance indicators consistent, or is the integrity of the underlying data suspect?

  In some cases, you may need to add other measures that substitute average figures, or industry standards, for unavailable or non-continuous values.

- Is the data at some of the lower drill-down levels too sparse to be useful? Should the model be redesigned, or should drill-through targets be added?

  Consider expressing some values as ranges rather than absolutes, to create useful groupings such as responsiveness or rates of return, for example, or to hide sensitive details, such as salaries.

- Could the data flowing from different databases, models, and reports be better coordinated, perhaps by using normalized measures, to ensure that computer resources are not overburdened?

- Has anyone developed a calculated column or exception dimension that could be added to the standard reports for the benefit of all?

If you maintain regular contact with your report users, you can incorporate their feedback into your model enhancements. If you change your model and cubes, use the label and description fields for each dimension, level, and measure, so that reports created from your model are clear and intuitive.

Upgrade an IBM Cognos Series 7 Model

To upgrade models created in earlier versions of Transformer, you must save them in Model Definition Language (MDL) format before you can import them into Transformer version 8.x. This ensures that equivalent definitions are created for all model objects. You can upgrade models from IBM Cognos Transformer, versions 7.x.

You can open an IBM Cognos Series 7 model with secured cubes in Transformer versions 8.x, and convert the IBM Cognos Series 7 user class views to IBM Cognos 8 custom views. You can then choose the authentication provider you want to use with the custom views. For more information, see "Adding Security" (p. 173).

During the transition from an IBM Cognos Series 7 namespace to an alternate security provider, you can use the PowerCube property **All applicable namespaces** to associate all applicable namespaces during migration testing. When you associate all the applicable namespaces to the cube, you can ensure that the group, role, or user dimensional filtering is consistent with that which had been applied for the IBM Cognos Series 7 user class. This option is supported only for migration testing, and cannot be used to deploy cubes in production environments.

You can change the association for an IQD data source to that of an IBM Cognos 8 data source, thereby taking advantage of the enhancements available when using an IBM Cognos 8 package or report data source. You can change the association for IBM Cognos Series 7 .iqd files and for Framework Manager .iqd (externalized query) files, after the updated model has been saved in Transformer 8.x. For more information, see "Change a Data Source Type" (p. 56).

When importing .mdl files from earlier versions, some features may not convert correctly, such as legacy data that contains special characters, spaces, and quotation marks. For more information, see the migration documentation delivered with your version of the product.

Tip: If you plan to upgrade, ensure you save all your models as .mdl files before you attempt to upgrade them.

Steps

1. Open the model in the earlier version of Transformer and, from the **File** menu, click **Save As**.

2. In the **Save as Type** box, click **Exported Model Files (*.mdl)**.

   Tip: By default, Transformer saves models in the My Documents/Transformer/Models directory. You can set the default location to which Transformer saves models by changing the **Models** directory setting on the **Directories** tab of the **Preferences** property sheet.

3. Open your new .mdl file in the current version of Transformer, make any required changes to the model design, and save it, again selecting the .mdl format.

Tip: If your IBM Cognos Series 7 model includes security, you will receive a message when you open the model in Transformer version 8.x indicating that you must choose how to manage the security during the upgrade process. For more information, see "Upgrade an IBM Cognos Series 7 Secured PowerCube" (p. 183).

When you are ready to use the model in your production environment, you may want to save it as a .py?-format file.

IBM Cognos Transformer version 8.x models (.mdl and .pyj files) are not backward compatible with Transformer versions 7.x. As a result, we strongly recommend that you maintain the .mdl file for the Transformer 7.x model for a period of time following an upgrade.

Chapter 3: Data Sources for Your Model

Models contain definitions of dimensions, levels, and measures. They also contain features such as calculated measures, dimension views, or custom views that you add to the basic PowerCube definition to meet your particular business intelligence needs.

By querying the data in the specified sources, you create the multidimensional PowerCubes or cube groups required by users of the IBM® Cognos® 8 Business Intelligence components, such as Analysis Studio.

You can store your models as text (.mdl) files or in binary format (.py? files, where ? is replaced by a character that represents the version of Transformer used to create the model). In Transformer versions 8.3 and 8.4, models are given a .pyj extension. Binary format (.py?) files are not compatible across different versions of Transformer. For this reason, we recommend that you always create a text (.mdl) version of your model.

Data sources can be one of the following:

- Structural (dimensional)

  Contain the columns that define the model structure, such as the categories in each dimension. Structural sources usually contain many columns and few rows.

- Transactional (fact)

  Contain the columns for the measures to be tracked. They usually contain many rows and few columns, typically one for each dimension and one for each measure.

- Mixed

  Contain the columns that define the model structure and the columns that contain the measures to be tracked, using the same data source.

Techniques for Designing Data Sources

When setting up the data sources for your model, consider the following principles:

- Where possible, design your data so that the structural information for each dimension is provided by one source.

- Ensure that each data source contains enough information to generate the categories for a dimension without database joins. If you must use database joins, join queries from separate database tables using tools such as Framework Manager, before you import the data.

- In addition to database security, be aware that different releases of Transformer offer different options for protecting your cubes and controlling access to information. For example, Transformer version 7.x supports user class views whereas Transformer version 8.x replaces this feature with custom views that can be associated with IBM Cognos 8 security objects (users, groups, and roles).

Data Source Types

Transformer version 8.x supports IBM Cognos Series 7 data sources as well as packages and reports that contain IBM Cognos 8 query items.

This section lists the supported data sources, summarizes the information you must specify for each data source, and identifies associated limitations.

Tip: You can also click the **Help** button, where available, for context-sensitive information about the parameters that you must specify.

Although you can add an unlimited number of data sources or columns to each model, you must perform any necessary joins between the various data files before you import the data into your Transformer model. You must also ensure that each data source contains sufficient information to provide the necessary context for any drill-down paths specified in the model.

IBM Cognos Package or Report

You can import query items from relational or dimensionally modeled relational (DMR) packages and reports, and the associated filters and prompts, by choosing the **Package** or **Report** data source type and browsing and selecting from the available metadata. In relational packages and reports, measures appear as defined in Framework Manager. Note that

- Transformer cannot view or override Framework Manager governor settings.
- Query items that are calculated appear as regular query items. You should review the rollup rules in Transformer when you use these calculated items as measures to avoid incorrect rollup results. For example, a rollup rule of **Sum** should not be applied to a measure that uses a calculated item when the value is expressed as a percentage. For more information, see "Rollup Functions" (p. 352).

Tip: Extra queries may appear when you import a report that contains prompt pages. These queries can be identified by the presence of query items named **Use Value** and **Display Value**. Avoid importing query items from these queries.

After import, you can combine the IBM Cognos 8 data with the data from other sources as required. Individual query items can be used as source columns in the Transformer model, and can be updated using the **Modify Columns** feature.

For information about using reports as a metadata source in Transformer, see "Create a Model" (p. 47).

For information about modeling IBM Cognos relational and DMR data sources, see the Framework Manager *User Guide*.

Dimensionally Modeled Relational Packages

When you access metadata from a dimensionally modeled relational package, you can import and leverage the dimensions, or import the query items or metadata that make up those dimensions. You can also import the measure metadata. Metadata from dimensionally modeled relational packages can be directly accessed in two ways:

- **Insert Data Source** option

Using the **Insert Data Source** option on the **Edit** menu, you can select query items and measures from a dimensionally modeled relational package as though it were a relational package. In other words, you will not see the dimensional structure. Use this option when you want to import the measures or specific query items from the package, but not the dimensions.

- **Insert Dimension from Package** option

  Using the **Insert Dimension from Package** option on the **Dimension Map**, you can select the dimensions, hierarchies, or levels that you want to import onto the Transformer version 8.x **Dimension Map**. Use this option when you want to include a small subset of dimensions or you do not want to use the **New Model** wizard to import dimensions.

If you want to take advantage of Transformer's relative time functionality, do not import the date dimensions from dimensional packages. Instead, use the **Insert Data Source** option to import the appropriate date field to create your time dimension.

OLAP Package

Transformer allows you to leverage metadata from other published OLAP packages. As a result, Transformer PowerCubes can be used as high speed data access cache methods for distributing smaller or focused areas of your business information.

Consider the size of the resulting cube when you use another OLAP package as a PowerCube data source. OLAP sources, such as Essbase, can include significant data that is not be appropriate for PowerCubes. However, taking a specific segment of data from these sources can be very useful, particularly if you intend to mix that data with other data sources for further reporting or analysis.

When you use OLAP sources to populate your Transformer models

- Import the dimensions that you require.

  SAP variable prompts are supported and should be used where necessary to limit the data to a specific segment of your data source. For more information, see "Work with SAP BW Data Using a Package in Framework Manager" (p. 361).

- Create the time dimension in the same way that you create fact queries.

  Transformer does not support importing time dimensions from any OLAP source, including PowerCubes. To create the Transformer time dimension with relative time categories, import your time information from either an IBM Cognos 8 relational package or report source, or from a flat file exported from IBM Cognos 8 or the original OLAP vendor.

In Transformer, you add dimensions from OLAP packages directly from the **Dimension Map**. This is a useful way to begin creating conformed dimensions and, to some extent, to reusing portions of the published metadata from the source dimension.

Using the **Insert Dimension from Package** option on the **Dimension Map**, you can select the dimensions, hierarchies, or levels that you want to import from any OLAP package on to the Transformer **Dimension Map**.

SAP BW Package

You can use Transformer to import both dimensional and fact data from an SAP BW query source. To do so, the SAP BW query package must be in a specific format. The Transformer PowerCubes you create with these specifically constructed SAP query packages can be used as high speed data access cache methods for distributing smaller or focused areas of your business information.

There are three stages to importing an SAP BW query to access both dimensions and facts using IBM Cognos 8:

- creating a query in SAP BW Business Explorer Query Designer
- creating a package in Framework Manager
- creating a model in Transformer

Business Viewpoint Studio

IBM Cognos 8 Business Viewpoint Studio helps to provide you with one version of the truth for dimensions used in an enterprise's performance management processes. Business Viewpoint Studio is a controlled, collaborative, workflow-oriented business process to manage both manual and automated changes to all data related to how enterprises analyze and manage their business. Business users are given the responsibility and authority to manage dimensions in their areas of domain responsibility. By using workflows, proposed changes and additions to dimensions are approved and validated before being distributed throughout the enterprise. You use the Business Viewpoint Client to subscribe to Business Viewpoint Studio master dimensional data from within Transformer.

Impromptu Query Definition File

Impromptu Query Definition (.iqd) files are generated from either IBM Cognos Impromptu or Framework Manager (as externalized query files). .iqd files are Transformer data sources that point to source databases specified in the cognos.ini file.

The **Series 7 IQD Bridge** component must be installed, and can be installed only on a supported IBM® Cognos® Series 7 version 7.4 platform. For more information, see "Series 7 IQD Bridge" (p. 19).

When importing an IQD data source in Transformer, accept the default **Isolation level** or specify an alternative. For more information, see "Isolation Levels for an IQD Data Source" (p. 64).

Tip: The Transformer version 8.x .ini file name and installation location have changed from the Transformer version 7.x file name and installation location. The Transformer preferences file name (previously the .ini file) is cogtr.xml, which is located in the *installation_location*\configuration directory. For information about accessing databases using the Series 7 IQD Bridge, see "Connecting to an IQD Data Source" (p. 63).

Delimited-field Text with Column Titles

Flat files are an excellent data source for achieving fast cube builds. Flat files are also recommended when you want to import OLAP fact data.

With **Delimited-field text with column titles,** input values are obtained from an ASCII text file with one record per line or row. The values in the first line represent column names.

When importing a flat file data source in Transformer, specify how the fields (column values) are delimited in the **Field delimiter** box, and either accept the default **Character set** or specify an alternative.

In Transformer version 8.x, you can specify **Unicode** as a valid character set. Using an IBM Cognos 8 report, you can define the fact query and the data you want to import for your PowerCube. You can then export the report to a .csv file that can in turn be used as the fact query data source in your Transformer model.

For more information, see "Character Sets Used with Delimited Text Data Sources" (p. 66).

Delimited-field Text

Flat files are an excellent data source for achieving fast cube builds. Flat files are also recommended when you want to import OLAP fact data.

With **Delimited-field text**, input values are obtained from an ASCII text file with one record per line.

When importing a flat file data source in Transformer, specify how the fields (column values) are delimited in the **Field delimiter** box, and either accept the default **Character set** or specify an alternative.

In Transformer version 8.x, you can specify **Unicode** as a valid character set. Using an IBM Cognos 8 report, you can define the fact query and the data you want to import for your PowerCube. You can then export the report to a .csv file that can in turn be used as the fact query data source in your Transformer model.

For more information, see "Character Sets Used with Delimited Text Data Sources" (p. 66).

Access Table

With an **Access table**, input values are obtained from a Microsoft Access file. Transformer uses the Microsoft ActiveX Data Objects (ADO) driver to access the data.

In Transformer, select a **Table** or **range** and a **Character set**, such as **DOS Code Page** or **Windows ANSI**.

Access Query

With an **Access query,** the source table is described in a Microsoft Access Query (.mdb file). Transformer uses either the Microsoft ActiveX Data Objects (ADO) driver to access the data, or runs the SQL queries stored in the .mdb file to get the source columns from an ODBC-enabled server database.

Note: Password-protected files are not supported.

In Transformer, select a **Table** or **range** and a **Character set**, such as DOS Code Page or Windows ANSI.

Excel crosstab

With an **Excel crosstab**, input values are obtained from an Excel crosstab file.

Transformer supports both .xls and Excel 2007 .xlsx file formats. You must have Excel 2007 or the 2007 Office System Driver data connectivity components installed on your computer to select the .xlsx file format when browsing for a data source.

For more information, see "Named Ranges" (p. 66).

Excel Database

With an **Excel database**, input values are obtained from a Microsoft Excel spreadsheet database file.

Transformer supports both .xls and Excel 2007 .xlsx file formats. You must have Excel 2007 or the 2007 Office System Driver data connectivity components installed on your computer to select the .xlsx file format when browsing for a data source.

In Transformer, select a range name from the **Table** or **range** box. For more information, see "Named Ranges" (p. 66).

PowerHouse Portable Subfile

With a **PowerHouse portable subfile**, input values are obtained from a Cognos® PowerHouse® 4GL portable subfile.

In Transformer, specify the portable subfile dictionary (.psd) file or the data (.ps) file. Accept the default **Character set** or specify an alternative. For more information, see "PowerHouse Data Source Parameters" (p. 68).

Fixed-field Text

With **Fixed-field text**, input values are obtained from an ASCII text file with one record per line. Each field starts at the byte immediately following the preceding field; the width of each field occupies a specified number of bytes. Each row ends with a text line delimiter.

In Transformer, manually add columns to your data source by specifying the position and length, in bytes, of each column in the source file. Accept the default **Character set** or specify an alternative.

Fixed Field and Record Without CR LF

With **Fixed field and record without CR LF**, input values are obtained from an ASCII text file. Each field starts at the byte immediately following the preceding field; the width of each field occupies a specified number of bytes. The record end is not marked by a text line delimiter.

In Transformer, manually add columns to your data source by specifying the position and length, in bytes, of each column in the source file. Accept the default **Character set** or specify an alternative.

Data Source Limitations

Some of the Transformer version 8.x data sources have known limitations, as described below.

IBM Cognos 8 Package or Report Data Source Limitations

IBM Cognos 8 Data Source Using Model as a Data Source

When an IBM Cognos 8 package or report data source uses a model, and the model is subsequently altered to remove query subjects, the Transformer model that uses that data source is not automatically updated to reflect the changes.

You can use the **Modify Columns** feature to detect and fix the changes. Otherwise, the changes in the source model are detected only when queries are executed, or when you run **Check Model** on the Transformer model.

Rollup Aggregate Settings

When you use columns from a data source that uses package-based measure query items as measures in Transformer, Transformer maps the rollup or regular aggregate setting of the package source query items to a corresponding measure rollup type in Transformer. However, when you use a query item from a data source that uses report-based measures as query items in Transformer, Transformer always creates the measure with the default rollup type **Default** (**Sum**) because the report metadata does not return any aggregate settings.

When you work with a data source that uses report-based measure query items, review the measure properties defined on that data source to ensure that the measure has the appropriate rollup type.

Aggregate Values for Imported Query Items May Not Match the Source Package or Report

When you import a Report Studio report, a Query Studio report, or an IBM Cognos 8 package as a data source, you will receive an error message if the report or package contains an aggregation rule that is not supported by Transformer. When an aggregation rule is not supported, Transformer defaults to the **Sum** rollup rule.

Data Preview

Data preview windows may not show data rows grouped or sorted as they appear in a report; however, this does not impact how Transformer uses the data when building cubes.

Extra Query Items

When you create a data source using an IBM Cognos 8 report that contains groupings, the report query sometimes shows extra query items.

The extra query items are created to support grouping, and should be ignored when selecting the data items for the Transformer query.

Prompts in Report Data Sources

You can use an IBM Cognos 8 report with prompts as a data source in Transformer. You must provide values for any mandatory prompts (p. 52) when adding a query based on the report data source to the model. Transformer asks you for these values only the first time you add a query from a report data source. Any values you provide are cached.

If you want to add a second query using the same report as a data source to your Transformer model, you will not be prompted to provide values for mandatory prompts. The values in the cache will be used. Although you can refresh the source when adding the second query to force Transformer to reprompt you for values, data will still be retrieved based on the first query.

To create two queries in your Transformer model that are based on the same report data source, where you want to provide different values for mandatory prompts, you must duplicate the report data source. Use one report data source to add the first query to the model and use the duplicate report data source to add the second query to the model.

IQD Data Source Limitations

IQD Data Source Support

IQD data sources require that the Series 7 IQD Bridge component be installed. Because the Series 7 IQD Bridge component can be installed only on a supported IBM Cognos Series 7 version 7.4 platform, IQD data sources are supported only on supported IBM Cognos Series 7 Version 4 platforms.

For an up-to-date list of supported environments, visit the IBM Cognos Resource Center Web site (http://www.ibm.com/software/data/support/cognos_crc.html).

IBM Cognos Series 7 Security

PowerCubes with IBM Cognos Series 7 Security

PowerCubes with IBM Cognos Series 7 security can be built on any IBM Cognos 8 platform, including Linux and HPUX Itanium, provided that the Content Manager is installed on a supported IBM Cognos Series 7 Version 4 platform.

For an up-to-date list of supported environments, visit the IBM Cognos Resource Center Web site (http://www.ibm.com/software/data/support/cognos_crc.html).

Designing Successful IBM Cognos 8 PowerCubes

The most successful business intelligence applications are designed with well planned models. This includes an analysis of how the data in the models will be used by report and analysis users. Consider the following concepts when designing PowerCube models for use in IBM Cognos 8.

Conformed Dimensions

Dimensions are conformed when the data values that come from the original data sources use the same business keys, or source data, that is used in other packages or models in your IBM Cognos 8 environment. Conformed dimensions allow your users to combine or cross data sources successfully when their business needs require that they do so.

For example, consider that your goal is to drill through to product line information between two reports. The first report is based on a PowerCube package, and the second report is based on a relational package. Each product line in the relational package should include a business key, or unique identifier. In the PowerCube model, the **Source value** for each category in the Product Line dimension should reference the same data value as the business key in the relational package.

When the same business keys and source values are used throughout your IBM Cognos 8 application data, end user success with reporting and analysis will increase substantially.

Conformed dimensions are also key in successful data analysis using multiple PowerCubes. When two cubes are to be used together, as with drill through, ensure that the dimensional structure and the category source values are the same in each cube model. Changes in the structure of a dimension in one cube, for example, by adding another level, will impact both the reports and drill-through applications that use the two cubes.

IBM Cognos 8 Business Keys

In IBM® Cognos® 8 Business Intelligence Reporting, PowerCube categories, or members, have business keys that can be used for advanced reporting or in drill-through scenarios. During Transformer model design, you can determine the IBM Cognos 8 business keys by setting the level source values.

Tip: Report Studio report authors can determine PowerCube business keys using a calculation such as

```
roleValue('business key',[mycube].[Product Dimension].[All
Products].[Product
Line])
```

Member Unique Names

In the IBM Cognos 8 Web studios, the Member Unique Name (MUN) is the unique identifier for locating the category or member in the data source. The MUN is much like the business key in a table.

IBM Cognos 8 metadata uses the Transformer model category codes when defining the MUN of a PowerCube category or member.

Member Unique Names are used

- as data item references for categories or members in any IBM Cognos 8 report specification
- as the value passed in PowerCube to PowerCube drill through in any IBM Cognos 8 report
- as identifiers for categories or members used in filters, expressions, parameters, or calculations in IBM Cognos 8
- to return categories or members to IBM Cognos 8 applications

  Any time an IBM Cognos 8 application requests the category or member, the MUN ensures that the unique category or member is returned.

Tip: You can view the Member Unique Name for a category or member in Report Studio. In Report Studio, open a PowerCube package, select a category, and view the category properties.

Each time a PowerCube category code changes, the MUN reflects the change. When categories or members are directly referenced in expressions, filters, or reports, and the MUN changes, the category or member is no longer found. This is because the original MUN is contained in the report specification.

Member Unique Names can change for different reasons:

- Changes in the hierarchy and level structures can result in changes to MUNs.
- Relative time categories may change, for example, when the current quarter moves from one to the next.
- If a source value changes, the category code used in the MUN also changes, unless the category code is specifically set to use a unique data item in the model design.
- The production environment may have more categories or members than the test environment.
- The category or member may no longer exist in the data source.

To avoid these problems, we recommend the following practices:

- Use unique codes and keys within a dimension for the category or member keys.

  Ensure that your Transformer model source values have unique values throughout the levels of each dimension. This ensures that the model category codes, and therefore the MUNs, are more stable.
- Use unique conformed source values for similar dimensions between the target and source environments when enabling drill through.
- Ensure that the business keys and dimension metadata structure are the same between the production and test environments.
- If the data source is an IBM Cognos 8 package or report, do not change the business keys in after going into production.
- Resolve the non-unique keys within a dimension in the data source.

  Tildes are not recommended in the category codes, as they can produce unstable MUN values.
- If you have tildes within your category codes, do not use the **Clean House** feature.

  Using the **Clean House** feature will most likely change the category codes.
- We strongly recommend that you keep a backup of your .mdl file and revert to the backup .mdl model file if the current model file becomes corrupt and requires a **Clean House** action.

Recommendation - Resolve Uniqueness Problems in Your Data Source

To avoid ambiguity problems in your reports, we recommend that you design your models so that no two categories in a level represent identically-named distinct categories, such as cities with the same name in two or more regions.

When you create models in Transformer, multiple non-unique categories imported into the same level are made unique by appending ~### to the duplicate codes, where ### represents an ascending numeric sequence.

The mappings between these assigned codes and their associated source values are stored in the Transformer model for use in subsequent cube build operations. However, errors may arise if the model is not saved after a cube refresh, or if the processing order changes for any reason.

For example, IBM Cognos 8 report specifications reference categories or members of an OLAP package, including PowerCubes, using a unique identifier referred to as Member Unique Name

(MUN). This MUN is generated for each category in a PowerCube and is based on the fully-qualified path of category codes in the dimension, according to where the category exists within the dimension. If the category codes change for any reason, the report specification can no longer locate the original MUN. The report author must modify the report to point to the updated category or member.

If your source data contains columns that populate levels that are not unique, an error message warns you of the potential problem when you attempt to generate categories. However, this prompting occurs only if the data source contains all the columns required for the levels in question. If categories for some levels have values derived from other transactional data sources, uniqueness conflicts may arise but remain undetected. Also, if you select an optimization setting that maximizes query speed, Transformer does not check your model for uniqueness conflicts. For this reason, we recommend that you save your model after every cube build.

To ensure successful business intelligence applications using IBM Cognos 8, we strongly recommend that the data sources that feed your Transformer models have unique business keys or source values through the levels of each structural dimension that you model. In addition, source values that conform with the business keys in other applications used in IBM Cognos 8 will have the best success rates when used with drill-through applications and other business intelligence applications.

Use Calculated Columns to Qualify Non-unique Data

You can use Transformer calculated columns, Framework Manager, or a query tool to render the level values unique, or simply redefine the data in your source so that it does provide unique values. Correlate the settings in the appropriate property sheets of your model so that your data is correctly mapped in the relevant IBM Cognos 8 components.

Ensure that the data sources that feed your Transformer models account for this uniqueness. Provide conformed values to any applications that you want to use with IBM Cognos 8 Business Intelligence products.

Create a Model

To create the cubes that you need for OLAP reporting, you begin by creating a model. This involves

- specifying the data sources for the model and any required security credentials
- defining dimensions, levels, and measures based on the selected query objects in your IBM Cognos 8 package or report, or the tables, rows, or columns of your other data sources
- defining cube objects that use the contents of the model to create PowerCubes or cube groups

You can have multiple data source queries associated with each package or report you create as a data source.

When the IBM Cognos 8 data source is a relational or dimensionally modeled relational package, you can import query items to create your structural or transactional queries.

You can create data source queries using IBM Cognos 8 reports created in Query Studio or Report Studio using relational or DMR packages. You cannot create data source queries using IBM Cognos 8 OLAP reports. Data source queries using IBM Cognos 8 reports perform most efficiently when the report is a list. Graphs, dashboards, crosstabs, and complex reports do not provide appropriate data to cube builds and therefore cannot be used for data source queries.

Do not import dimensional packages using the **Insert Data Source** command, if you want the dimensional structure maintained. Use the **Insert Dimension from Package** command instead.

When using an IBM Cognos 8 report created in Query Studio or Report Studio, or an IBM Cognos 8 package as a data source, you may be asked to provide values for existing prompts. For more information about prompt support, see "Edit Existing Prompts in IBM Cognos 8 Reports and Packages" (p. 52).

When you create a data source using an IBM Cognos 8 report, and the report includes data in multiple languages, some of the language characters do not display properly in the Transformer **Data Source Viewer**. These characters are displayed as -- .

Transformer does not support reports with multilingual data as a data source. When the operating system locale is properly set, Transformer displays the characters for that locale.

SAP BW Packages

SAP BW packages can be used to import fact and dimensional data. For information about preparing SAP queries and creating packages in Framework Manager for use in Transformer, see "Guidelines for Working with SAP BW Data for Use in Transformer" (p. 361). For information about creating an SAP package-based model, see "Steps to Create a Model Using an SAP BW Package" (p. 50).

Steps to Create a Data Source Query Using an IBM Cognos 8 Package or Report Data Source

1. From the Transformer Welcome page, click **Create a new model.**

   Tip: If you are already in Transformer, click **New** from the **File** menu to open the **New Model** wizard.

2. Type a name for your new model and click **Next.**

3. In the **Data source name** box, enter the name of the IBM Cognos 8 data source and, in the **Data source type** box, select **Package** or **Report.**

   Tip: If you want the data source name to default to the name of the package or report that you select in the next step, leave the **Data source name** box blank.

4. Click **Browse** to open the metadata browser and select a package or report from the available list.

   Tip: You can also click the drop-down arrow to select a recently used package or report from the list.

5. In the **Browse Metadata** dialog box, select the package or report to use for the data source, and click **OK.**

6. Click **Next.**

7. If you select a report as a data source and it contains mandatory prompts, provide values for the prompts.

   Transformer cannot execute queries if values are not provided for mandatory prompts. For information about prompts, see "Edit Existing Prompts in IBM Cognos 8 Reports and Packages" (p. 52).

8. In the **Query name** box, type a name for the new query.

   Note: When you use the **New Data Source** wizard to create a new data source, you create one query at a time.

9. In the **Source** list, select the query items to import and click **Add** to add the selected query items to the Transformer query.

   If the data source is a report and the report contains multiple queries as in the case of some Report Studio authored reports, each query will be shown with its relevant query items.

   Tips:

   - Report Studio authored reports will show queries that are associated with the list. The relevant query items for those queries will be available for use in Transformer.

   - If the data source is a package with dimensions, you can import the dimensions using the **Insert Dimension from Package** option.

10. Click **OK**.

11. If you want Transformer to automatically create a preliminary dimensional structure for you, on the last page of the **New Model** wizard, select the **Run AutoDesign** check box.

12. Click **Finish**.

13. If you are prompted for data source connection and signon information:

    - Select the connection and click **OK**.

    - Choose whether to enter a valid user name and password for the current session, or create a Transformer signon for the current and subsequent sessions:

      When you enter a valid user name and password and click **OK**, the signon will be used only during the current Transformer session. The signon will not appear in the **Signons** list, and is not saved in the model.

      To create a Transformer signon that appears in the **Signons** list and is saved in the model, enter a valid user name and password and select the **Create a Transformer signon from the user name and password or select an existing one for use with this data source** check box. In the **Transformer Signon** box, click the drop-down arrow to select an existing signon, or click the **Add** button to open the **Signon** dialog box to create a new signon. In the **Signon name** box, type the signon name and specify whether to prompt for a password and click **OK**. If you do not select the **Prompt for password** check box, in the **Confirm Password** dialog box, re-type the password and click **OK** twice.

    For information about creating Transformer signons, see "Create a Signon" (p. 80).

14. If you want to add another query to the package or report data source, click **Yes** when prompted to add another query, and repeat steps 8 to 9.

15. Save the model.

    Tips:

- By default, Transformer saves models in the My Documents/Transformer/Models directory. You can set the default location to which Transformer saves models by changing the **Models** directory setting on the **Directories** tab of the **Preferences** property sheet on the **File** menu.

  On Windows Vista, Transformer saves models in the Documents/Transformer/Models directory.

- When prompted to save in binary (.py?) or text (.mdl) format, use the latter when exporting models or, to avoid possible fragmentation problems, when you have made a lot of changes since your last save action.

- In addition to saving your models in .mdl format, you should regularly use the **Check Model** command from the **Tools** menu to help you diagnose and resolve any problems in your model design.

- In Windows, the .pyj model file extension is not automatically associated with Transformer version 8.x. To open a model in Transformer by double-clicking the .pyj file, you must first create the association in Windows.

Use the **Data Source Viewer** to view sample data and, for supported data sources, the Cognos SQL or the native database SQL.

Steps to Create a Model Using an SAP BW Package

1. In Transformer, click **Create a new model**.

2. In the **New Model Wizard**, click **Cancel**.

3. With the **Dimension Map** pane selected, from the **Edit** menu, click **Insert Dimension from Package**.

   You use the **Insert Dimension from Package** wizard because it

   - creates a single query for each dimension and for the facts.

   - imports facts and dimensions in the same manner as dimensionally-modeled relational models. That is, facts and dimensions are imported at the same time.

   - ensures that the scope is set properly between the dimensions and facts.

   - populates the dimension with the appropriate business key and caption information.

   - only imports the necessary items from the BW package required for cube building, when the metadata is imported. This reduces the number of attributes and keeps the data volumes to only the necessary items for cube building.

4. Click **Browse** to open the metadata browser.

5. In the **Browse Metadata** dialog box, select the package that contains your SAP BW query and click **OK**.

6. In the **Insert Dimension from Package** dialog box, click **Finish**.

7. In the **Select Dimension and Measures from Package** dialog box, click the dimensions and measures to include in the data source.

   Select a query item that will provide the dates for the PowerCube. Note that the dates for the PowerCube can be derived entirely from the transaction data.

8. If there are errors or warnings, you are notified. In the **Data Sources** pane, expand the package to view the data source queries and query items. Key figures or measures appear in the **Measures** pane.

   Ensure that the aggregation rule for each measure is correctly defined within Transformer to align as closely as possible with the aggregation rule defined in SAP BW.

   It is recommended that the storage type for all measures be set to 64-bit floating point.

   For the root level of each characteristic (dimension), ensure it is marked as unique.

   SAP BW presentation hierarchies may contain ragged paths, typically in association with the "not assigned" and "#" nodes in the hierarchy. The gaps in these hierarchies produce blanks at the associated level in the Transformer hierarchy. In Transformer, it is possible to define the text that should be used for blanks (the default text is "<blank>"). A best practice is to define a more appropriate text for blank entries for all such levels.

9. If you want to add another query, repeat steps 3 to 7.

10. Save the model.

Steps to Create a Model Using Other Data Sources

1. From the Transformer Welcome page, click **Create a new model** and click **Next**.

   Tip: If you are already in Transformer, click **New** from the **File** menu to open the **New Model** wizard.

2. Type a name for your new model, and click **Next**.

3. In the **Data source name** box, type the name of the data source and, in the **Data source type** box, select one of the available options.

4. Click **Next** to specify information about the data source.

   The parameters depend on the data source type that you selected on the previous page.

   If you selected an IQD data source, set the **Isolation level**. When you use .iqd files generated from IBM Cognos Impromptu, or externalized query files from any version of Framework Manager, ensure that the **Series 7 IQD Bridge** component is installed. This component must be installed on IBM Cognos Series 7 version 7.4 supported platforms. For more information, see "Isolation Levels for an IQD Data Source" (p. 64).

   For a Microsoft Access or Excel database, a table name or a named range of cells from the Excel worksheet must be specified.

   For more information about the parameters required for each data source type, see "Data Source Types" (p. 38).

5. Click **Browse** to open the data source browser and select a data source from the available list.

6. Click **Next.**

7. Specify whether or not to **Run AutoDesign**, and click **Finish**.

8. Confirm that your selected items appear as expected in the **Data Sources** list.

   Tip: To view sample data or the SQL from your data source, from the **View** menu, click **Data Source Viewer**.

9. Save the model.

   Tips:

   - By default, Transformer saves models in the My Documents/Transformer/Models directory. You can set the default location to which Transformer saves models by changing the **Models** directory setting on the **Directories** tab of the **Preferences** property sheet on the **File** menu.

     On Windows Vista, Transformer saves models in the Documents/Transformer/Models directory.

   - When prompted to save in binary (.py?) or text (.mdl) format, use the latter when exporting models or, to avoid possible fragmentation problems, when you have made a lot of changes since your last save action.

   - In addition to saving your models in .mdl format, you should regularly use the **Check Model** command from the **Tools** menu to help you diagnose and resolve any problems in your model design.

   - In Windows, the .pyj model file extension is not automatically associated with Transformer version 8.x. To open a model in Transformer by double-clicking the .pyj file, you must first create the association in Windows.

If you imported measures that are not in scope for a particular dimension, or that apply to more dimensions than your report users need, remove the extra items. Alternatively, ensure that the scope is set correctly between the dimensions and the fact query before you proceed.

Edit Existing Prompts in IBM Cognos 8 Reports and Packages

You can use an IBM Cognos 8 report created in Query Studio or Report Studio, or an IBM Cognos 8 package as a data source for a model (p. 47). These data sources may contain prompts that add interactivity for users. Prompts are questions that help users customize the information to suit their own needs. For example, a prompt may let users select a product type. Only products belonging to the selected product type are retrieved and shown in the report.

Prompts are either mandatory or optional; mandatory prompts require values before Transformer can execute queries. For optional prompts in IBM Cognos 8 report or package data sources, Transformer does not require that you provide values.

When using a report with prompts as a data source, the user interface presented by Transformer for prompts may be different than the user interface presented when running the report in the IBM Cognos studio. For example, a report with a prompt page defined to contain a single-value select drop-down list may be presented in Transformer as a multi-value select list. This behavior occurs because Transformer does not process any information from report prompt pages. Ensure that you understand the purpose of each prompt when using a report as a data source.

Many types of prompts exist, such as a text box prompt where you type a single value, and a value prompt where you select a value from a list. The prompt type determines the number of values you must provide, and the user interface shown for entering those values. For example, for a single-value prompt, Transformer uses the value when executing the query. For prompts with more than one value, Transformer lets you define multiple values. These values are presented in a drop-down list from which you can select more than one value when Transformer executes the query. Note that although default values for multi-valued prompts are supported in Transformer when using a report as a data source, the default values do not appear. However, if you do not provide any values for the prompt, the default values are used.

Report or package authors determine whether prompts are mandatory or optional. You cannot change this designation within Transformer, or create prompts using Transformer.

To create two queries in your Transformer model that are based on the same report data source, where you want to provide different values for mandatory prompts, you must duplicate the report data source. Use one report data source to add the first query to the model, and use the duplicate report data source to add the second query to the model.

Transformer may reflect a cascade prompt as two multi- or single-select prompts:

- The first prompt represents the first level of the original cascade prompt.
- The second prompt represents the second cascade level.

For query results in Transformer, only the second level is relevant. You should be aware that the first prompt may appear as a multi-select prompt in Transformer, even though the prompt in Report Studio is a drop-down list box; in this situation, Transformer will ignore the first prompt.

When using a report that includes prompts as a data source, some types of prompts produce supporting queries. Although these queries can be selected in Transformer, error messages will occur if you select them. You can ignore the messages.

Steps

1. Right-click a query in the model and click **Edit Prompts**.

   If the query does not contain any prompts, Transformer displays a message indicating that no prompts were found.

2. For each prompt that you want to specify values for, in the **Query Prompts** dialog box, click the **Value** box for the prompt that you want to define.

   - An asterisk beside a prompt name identifies a mandatory prompt.
   - A cross beside a prompt name identifies an obsolete prompt. This prompt exists in the Transformer query, but cannot be found in the IBM Cognos 8 report or package used as a data source.

- An arrow that points to the right beside a prompt name identifies a new prompt. This prompt exists in the IBM Cognos 8 report or package used as a data source, but cannot be found in the Transformer model.

3. Define the variable or value for the prompt:

   Tips:

   - For a prompt that requires a single value, such as a text box prompt, type the value that you want used when Transformer executes the query.

   - For a prompt that permits multiple values, such as a value prompt, the **Enter values for** dialog box appears. Type a value in the **Provide a value** box. Click the right arrow to add the value to the **Choices** list. Continue to type and add values until all values you want the user to choose from appear in the **Choices** list. Click **OK**.

   - For a prompt that requires a value selected from a hierarchy, such as a Tree prompt, the **Select values for** dialog box appears. Click a value and click **OK**.

   - For a prompt that requires a value selected from a list defined within the report or package, such as a list of cities, the list of defined values appears automatically. Click the value in the list.

   - For a prompt that requires a value selected within a specific range, type the initial value in the **range** (**Range-From**) box and the final value in the **range** (**Range-To**) box.

     - To quickly delete the value for one prompt, click the row in which the prompt appears and click **Clear Value**.

     - To quickly delete the values for all prompts, click **Clear All**.

     - When you enter a prompt value, the data type is not verified. Entering 15oo, rather than 1500, for a numeric value does not generate an error or warning even though the value is invalid. PowerCube creation may fail if you use an invalid value.

4. For each prompt that you want to delete from the query, in the **Query Prompts** dialog box, click the row in which the prompt appears and click **Remove**.

5. Click **OK**.

Generate a File of Prompt Values for Use in the Command Line

You can generate an XML command file that includes prompt values for one or more queries. This file is necessary to provide prompt values in UNIX environments. For more information about command line options and this command file, see "Command Line Options" (p. 237).

For each query that requires prompt values, ensure that your selections are up-to-date with the underlying package or report. (p. 55)

Steps

1. Right-click a prompt and click **Create Prompt Specification** to create a command line file for one query or click **Create Prompt Specifications For All Queries** to create a command line file for all queries.

2. Choose a folder for the command line file, type a name for the command line file, and click **Save** to save the command line file to your computer.

Remove Obsolete Prompts

Transformer stores prompts and their values in the Transformer model and uses this information for data requests and to generate a prompt specification. A prompt specification is supplied at the command line in batch mode to override the values stored in the model.

Prompts that are no longer in a package or report are identified as obsolete. This will occur, for example, if you remove a prompt for the report on which the model is based and save the report. Then, when you use the **Edit Prompts** command, that prompt is identified as obsolete in the **Query Prompts** dialog box. A prompt may also be identified as obsolete simply because the underlying report or package is unavailable. Once the report or package becomes available, the prompt becomes up-to-date again. **Tip:** If all or several prompts are marked obsolete, verify the connection to the server.

To ensure that your prompt selections are up-to-date with the underlying package or report, for each data source in the model, you must remove any obsolete prompts.

Steps

1. In the **Data Sources** window, right-click the query and click **Edit Prompts**.

2. In the **Query Prompts** dialog box, in the **Name** list, click any obsolete prompts.

   Tip: Obsolete prompts are indicated by an icon.

3. Click **Remove**.

   You must repeat these steps for each query.

Note: Alternatively, you can remove obsolete prompts by directly editing the prompt specification file that is generated from the model.

Specifying a Segmenting Prompt for an SAP BW Query

A segmenting prompt is used when querying a SAP BW data source for fact data. Also known as a BEx variable, a segmenting prompt ensures that the query retrieves a representative sampling of the fact data.

A segmenting prompt can be single value, multiple value, or a range. If you specify a range, it must be inclusive, including a value for both the beginning and end of the range. A segmenting prompt must be optional and have no default value specified.

Multiple prompts, or BEx variables, are allowed. If you have multiple prompts, you can select one as the segmenting prompt. The segmenting prompt should not have values specified in any query. Mandatory prompts that are not specified as the segmenting prompt must have a value specified. Optional prompts that are not specified as the segmenting prompt may or may not have a value, as necessary.

Since prompts apply to all queries in the Transformer model but are maintained separately for each query, the specified values should be the same in all queries. If there is more than one fact query, each may use a different prompt as the segmenting prompt. The queries can use the same segmenting prompt, if suitable.

Tip: Segmentation is only supported for fact queries that have the **Use stream extract** option selected on the **Data Source** property sheet, **Source** tab.

Steps

1. In the **Data Sources** window, right-click the query item named **Key Figures**.

   Key Figures contains measures.

2. In the **Data Source** dialog box, click the **Source** tab.

3. Ensure that the **Use stream extract** check box is selected. Click **OK**.

4. Right-click **Key Figures** again and click **Edit Prompts**.

   The **Name** column in the **Query Prompts** dialog box lists all the prompts for the data source.

5. In the **Prompt for segmenting data** list, select the prompt for segmenting the fact data.

   Only valid prompts are listed.

6. Ensure that the segmenting prompt does not have a default value specified.

   Tip: To clear the values for a prompt, click the prompt in the **Current prompt values** list, and click **Clear Value**.

7. Ensure that any mandatory prompts listed in the **Current prompt values** list have a value specified.

8. Click **OK**.

9. When prompted to apply the prompt values to all the queries in the package, click **Yes**.

   595074, 505076

Change a Data Source Type

If you decide after you import a data source in Transformer that the data can be more effectively sourced from a different data source type, you can change the data source.

For example, suppose you used a fixed-field text file as the data source for product returns information. Later, you determine that the same product return information is used in an IBM Cognos 8 report. You decide to change the data source for the product return information from the fixed-field text file to the IBM Cognos 8 report. You can change your IQD data sources in upgraded models to use an IBM Cognos 8 package or report, provided the same query items are available. The query item names need not be identical because, with the **Change Source Type** feature, you can map the query items from the .iqd file to the IBM Cognos 8 package or report data source that you create. This means that your models are no longer dependant on the administrator for changes to, or maintenance of, the externalized query.

IQDs generated by Framework Manager will continue to be supported in this IBM Cognos 8 release, but will not be enhanced. We will deprecate support for IQDs generated by Framework Manager in the next major release of IBM Cognos 8. As a result, we recommend that modelers publish the queries that are the basis for Transformer queries as part of published packages. You can then use the published query subjects to create Query Studio or Report Studio list reports, which can in turn be used to create a Transformer data source. Doing this allows you to become more self-sufficient as a modeler: you now have the ability to edit and maintain the report data source separate from the original package.

If the externalized query is based on SAP key figures, we recommend that you author an IBM Cognos 8 scheduled report in exported .csv file format that can be saved to a disk and made available to the Transformer model. You can include this as part of your normal batch script scheduling for cube builds.

You cannot change a data source for a query in the model to match another query already in the model.

You cannot add or remove columns when you change a data source type.

In the data source you want to change, all columns must match references in the new data source; you cannot match only some of the columns in the original data source.

Steps

1. In the **Data Sources** list, right click the data source you want to change, and click **Change Source Type**.

2. In the **New Data Source** wizard, in the **Data source type** box, click the drop-down arrow to select the new data source type.

   You cannot change the data source name.

3. Enter the required information about the new data source type:

   - For an IBM Cognos 8 package or report data source, click **Browse** to open the metadata browser and select a package or report from the available list.

   - For other data source types, specify the required information about the data source, or click **Browse** to open the data source browser and select a data source from the available list.

   If you click **Cancel**, the entire change data source type action is cancelled.

4. Click **Finish**.

5. In the **Query Definition** or **Modify Columns** dialog box, match the columns in the original data source to the columns in the new data source:

   - In the **Source** list, select a column.

- In the **Model** list, select the corresponding column.
- Click **Match**.

  The **Add** and **Remove** buttons are disabled.
- When you finish matching all the columns in the original data source to columns in the new data source, click **OK**.

  The **OK** button is enabled only after all the columns in the original data source are matched to columns in the new data source.

Create Dimensions from the Dimension Map Using OLAP and DMR Packages

The **Insert Dimension from Package** option is available only from the Transformer **Dimension Map**. This feature provides for dimensions to be created efficiently to resemble the original dimensional structure from your OLAP or DMR package as closely as possible.

When the data source is an IBM Cognos 8 OLAP package, you must import the dimensions from the package directly from the **Dimension Map**. You subsequently import the measures from any other data source Transformer supports.

Tip: If you want to use the measures from an OLAP source, we recommend that you create a relational query against the source data used to build the original OLAP source. This approach will provide the best possible performance.

Transformer version 8.x does not support parent-child hierarchies in OLAP data source packages. If you attempt to create data sources using OLAP packages with parent-child hierarchies, Transformer will interpret each parent and child as unique dimensions.

Steps to Create Dimensions from the Dimension Map Using OLAP and DMR Packages

1. In a Transformer model, right-click the **Dimension Map** and click **Insert Dimension from Package**.
2. Browse to select a dimensional package and click **Finish**.
3. In the dimension tree for the package, select the dimensions, hierarchies, or levels that you want to add to your model and click **OK**.

   The dimensions that you selected are added to the **Dimension Map**, and a package data source is added to the **Data Sources** list. A single query for each dimension is added under the package data source.

   Where possible, Transformer includes the relevant label and business key as the source column for each level in the imported dimension. Because different OLAP sources behave differently, you may need to change the query items used in the source or category code columns for the levels to ensure the dimension is in scope with the measures for the model.

   The import may include more metadata than expected. This allows you to refine the columns that are used within the dimension.

Tip: To ensure conformed dimensions across different packages, the original OLAP source should use unique business keys. This will help to ensure that Transformer generates category codes that are conformed to the OLAP source from which the dimension was created.

Add a Data Source to an Existing Model

Because the **New Model** wizard lets you specify only one data source, including one query at a time for each package or report data source, we recommend that you use the wizard to add the structural hierarchy (dimensional data) that is important to your business.

If your data source is an IBM Cognos 8 package or report, you can use the **Add Query From Package or Report** option to add an additional query to the data source. For all other data source types, you can use the **New Data Source** wizard to add all the other data sources required for your model.

As the modeler, you must ensure that each data source has a unique name and specify, if true, that all your category codes and source values are unique. This precaution significantly reduces

- the possibility that end user reports will be negatively impacted by changes in category codes
- drill-through problems
- other errors that can arise in distributed production environments

For more information, see "Recommendation - Resolve Uniqueness Problems in Your Data Source" (p. 46).

Tip: If you imported your dimensions to the **Dimension Map** from a package using the **Insert Dimension from Package** option, you can add another dimension from that package to the **Dimension Map** using the same option. This adds the additional query to your package data source and completes the **AutoDesign** process on the **Dimension Map**. If you add a query to an OLAP package data source using the **Add Query From Package or Report** option, you will need to manually construct the dimension.

Steps for an IBM Cognos 8 Data Source

1. Right-click the empty space in the **Data Sources** list for your model and click **Insert Data Source**.

   Tip: If the **Insert Data Source** option is unavailable, you may have right-clicked a package or report data source.

2. In the **Data source name** box, enter the name of the IBM Cognos 8 data source and, in the **Data source type** box, select **Package** or **Report**.

   Tip: If you want the data source name to default to the name of the package that you select in the next step, leave the **Data source name** box blank.

3. Click **Browse** to open the metadata browser and select a package or report from the available list.

   Tip: You can also click the drop-down arrow to select a recently used package or report from the list.

4. Click **Next.**

5. In the **Query Name** box, type a name for the new query.

   In the **New Data Source** wizard, you create one query at a time.

6. In the **Source** list, select the query items to import and click **Add** to add the selected query items to the Transformer query.

7. If you want Transformer to create preliminary dimensions in the **Dimension Map**, select the **AutoDesign** check box.

8. Click **Finish.**

   You will be prompted to add another query. If you want to add another query, click **Yes** and repeat steps 6 and 7. If you click No, a new data source containing the specified metadata appears. The source columns appear in the **Data Sources** list and, if you selected **Run AutoDesign**, a preliminary dimensional structure appears in the **Dimension Map**.

   Tip: Use the **Data Source Viewer** to view sample data and, for supported data sources, the Cognos SQL or the native database SQL.

Steps for Other Data Sources

1. Right-click the empty space in the **Data Sources** list for your model and click **Insert Data Source**.

   Tip: If the **Insert Data Source** option is unavailable, you may have right-clicked a package or report data source.

2. In the **Data source name** box, type the name of the data source and, in the **Data source type** box, select one of the available options.

3. Click **Next** to specify information about the data source.

   The parameters depend on the data source type that you selected on the previous page.

   If you selected an IQD data source, set the **Isolation level**. When you use .iqd files generated from IBM Cognos Impromptu, or externalized query files from any version of Framework Manager, ensure that the **Series 7 IQD Bridge** component is installed. This component must be installed on IBM Cognos Series 7 version 7.4 supported platforms. For more information, see "Isolation Levels for an IQD Data Source" (p. 64).

   For a Microsoft Access or Excel database, a table name or a named range of cells from the Excel worksheet must be specified.

   For more information about the parameters required for each data source type, see "Data Source Types" (p. 38).

4. Click **Browse** to open the data source browser and select a data source from the available list.

5. Click **Next.**

6. Specify whether or not to **Run Autodesign**, and click **Finish.**

7. Confirm that your selected items appear as expected in the **Data Sources** list.

Tip: To view sample data or the SQL from your data source, from the **View** menu, click **Data Source Viewer**.

Add Additional Queries to Existing IBM Cognos 8 Data Sources

IBM Cognos 8 package and report data sources are shown in the **Data Sources** list as a single data source with multiple associated queries. When you use the **New Model** wizard or **New Data Source** wizard, you can add only one query at a time.

To add a second query to your IBM Cognos 8 package or report data source, use the **Add Query From Package or Report** option. **Add Query From Package or Report** opens the **Query Definition** dialog box, where you can create the new query. When you create the new query, it is imported as an additional query under the package or report data source.

Tip: If you imported your dimensions to the **Dimension Map** from a package using the **Insert Dimension from Package** option, you can add another dimension from that package to the **Dimension Map** using the same option. This adds the additional query to your package data source and completes the **AutoDesign** process on the **Dimension Map**. If you add a query to an OLAP package data source using the **Add Query From Package or Report** option, you will need to manually construct the dimension.

You should not import query items from different query subjects unless the query was intended for or works well with the data for that scenario. Adding query items from multiple query subjects in a single import should be avoided, as it can result in cross-joins or queries that are not valid.

When adding queries from an IBM Cognos 8 report, Transformer displays the source query using the **Query name** specified in the query definition (for example, Query 1.0). Transformer does not display the query name as defined by the **Query Name** property in Query Studio or Report Studio.

Steps

1. In the **Data Sources** list, right-click a package or report data source, or a query under the package or report data source and click **Add Query From Package or Report**.

   Tip: If **Add Query From Package or Report** is not available, you may have right-clicked the area outside the data source.

   The **Query Definition** dialog box opens, showing the metadata from your source package or report data source.

2. In the **Query name** box, type a name for the new query.

   You create one query at a time.

3. In the **Source** list, select the query items to import and click **Add** to add the selected query items to the **Query definition details** box.

4. Click **Finish**.

   A new data source containing the specified metadata appears under the original data source. The source columns appear in the **Data Sources** list.

   Tip: Use the **Data Source Viewer** to view sample data and, for supported data sources, the Cognos SQL or the native database SQL.

Synchronize Columns with Your Data Source

As the modeler, you must ensure that the columns in your model reflect the current state of your data sources. There is limited error-handling. For example, an error message is issued if you try to add a query item that is already referenced in your model.

You can quickly resynchronize your model columns after a data update. You can add data source items to the model or remove columns from the model. You can also resolve mismatches between data source items and columns in the model. For example, if you rename data source items, Transformer may no longer be able to match those items with their corresponding columns in the model.

You can choose to resolve mismatches manually or let Transformer attempt to automatically resolve mismatches. For IBM Cognos 8 data sources, Transformer presents options for matches for your review. If Transformer determines that mismatches are caused by a structural change in the data source, Transformer provides a list of locations in the data source that may be possible matches for mismatched columns. If none of the locations provided are appropriate, or if Transformer is unable to suggest any locations in the data source, Transformer provides a list of items in the model that are similar to the source reference of the column. For each possible match, Transformer shows how closely the references match by using a percentage. If none of the items are appropriate, you can leave columns unmatched.

You can quickly resynchronize your model columns after a data update if your model uses a single query based on a text data file, an IQD, or an IBM Cognos 8 package or report. Resynchronize your model by invoking the **Modify Columns** command from the **Tools** menu.

Steps for Text Data Files or IQDs

1. In the **Data Sources** list, select the query whose columns you want to modify, and from the **Tools** menu, click **Modify Columns**.
2. For data source items that do not appear as columns in the model, select the items in the **Source** list and click **Add**.
3. For columns that you want to remove from the model, select the columns in the **Model** list and click **Remove**.
4. For unmatched columns identified by a plus sign (+) in the **Matched to Source** column in the **Model** list, do one of the following:
   - To manually match query items to columns, select a data source item in the **Source** list and a column in the model, and click **Match**.
   - To allow Transformer to automatically match query items to columns, click **Auto Match**. Review any messages that appear and click **OK**.

Data source items are now synchronized with the model.

Steps for IBM Cognos 8 Data Sources

1. In the **Data Sources** list, select the query whose columns you want to modify, and from the **Tools** menu, click **Modify Columns**.

2. If there are columns in the model that cannot be matched to data source items, you will get a warning message. Click **No** to keep these unmatched columns in the model.

   If you click **Yes**, Transformer deletes the unmatched columns.

3. For data source items that do not appear as columns in the model, select the items in the **Source** list and click **Add**.

   Tip: Click **Refresh Source** to refresh the source list for the data source. Click **Validate** to check whether columns in the model violate any Framework Manager governor settings.

4. For columns that you want to remove from the model, select the columns in the model list and click **Remove**.

5. For unmatched columns identified by and **X** in the **Matched to Source** column in the model list, do one of the following:

   - To manually match query items to columns, select a data source item in the **Source** list and a column in the model, and click **Match**.

   - To allow Transformer to automatically match query items to columns, select the columns in the model and click **Auto Match**.

6. If Transformer provides one or more locations in the data source that may be appropriate for unmatched columns, do one of the following:

   - If one of the locations is an appropriate match, select the location and click **Next** or **Finish**.

   - If none of the locations are an appropriate match, click **Match by reference instead** and click **Next**. From the **Candidates** list of data source items, click the one that you want to match to the column or click **Leave unmatched**.

7. Repeat step 6 for each mismatched column that Transformer attempts to resolve.

8. For any mismatched items that Transformer cannot suggest locations for, Transformer presents possible name changes for your review. From the **Candidates** list, click the item that you want to match to the column to or click **Leave unmatched**.

Data source items are now synchronized with the model.

Connecting to an IQD Data Source

You can use Impromptu Query Definition (IQD) files to access data from local or server-based databases, even if you do not have IBM Cognos Impromptu installed on your modeling computer. You can also use .iqd files with queries defined in Framework Manager. These are sometimes referred to as externalized queries.

Either of these .iqd file types requires that the **Series 7 IQD Bridge** component be installed. This component must be installed on IBM Cognos Series 7 version 7.4 supported platforms.

To review an up-to-date list of environments supported by IBM Cognos products, visit the IBM Cognos Resource Center Web site (http://www.ibm.com/software/data/support/cognos_crc.html).

Using IQDs from Previous Versions of Framework Manager

Framework Manager IQDs, or externalized queries, have the same limitations that existed in IBM Cognos Transformer version 7.x, including limitations related to local processing, multi-select statements, and multiple database connections in a single query.

In Transformer version 8.x, however, you can use the query items from an IBM Cognos 8 package or report. These data sources do not have the same limitations as the Framework Manager IQD.

If you have externalized queries as a data source, we strongly recommend that the query subjects be published in a package, and the query items be imported in Transformer version 8.x using the package as a data source. As an alternative, you can create a report based on the package and then create a data source using the report as a data source. This method provides functionality that was unavailable in externalized queries in previous releases, such as support for multi-select statements, local processing, and multi-database connections. For more information about using IBM Cognos 8 packages as a data source, see "Create a Model" (p. 47).

When you create a report based on this published package, the Transformer modeler can update the report whenever changes are required. The Transformer modelers are therefore no longer dependant on the modelers for revisions to the externalized query for the purposes of Transformer modeling.

For compatibility between IBM Cognos 8 version 8.x and releases prior to version 8.3, you must specify your IQD data source connection information in the Cognos.ini or cs7g.ini file. The cs7g.ini file is located in *installation_location*/c8/cs7Gateways/bin directory.

If you are working with a UNIX data source, ensure that the server connection information in the INI file on the Windows computer on which you created the model matches that stored on your UNIX servers. Otherwise, the source database cannot be read.

Isolation Levels for an IQD Data Source

When you create a data source in Transformer using an IQD, you must specify an isolation level, typically the same one as was used when the .iqd file was created. This is necessary so that the data is processed properly in the model.

If you must change the **Isolation level**, you can do so when you create your data source using the **New Data Source** wizard, or later, from the **Data Source** property sheet.

The following table describes the IQD isolation levels available for selection in Transformer.

Isolation level	Description
Default	Uses the isolation level that was originally specified in IBM Cognos Impromptu when the .iqd file was created.
Read Uncommitted	Makes changes made by other transactions immediately available to a transaction.
Read Committed	Allows a transaction access only to rows that were committed by other transactions.

Isolation level	Description
Cursor Stability	Prohibits other transactions from updating the row on which a transaction is positioned.
Reproducible Read	Ensures that rows selected or updated by one transaction are not changed by another transaction until the first transaction is complete.
Phantom Protection	Prohibits access by a transaction to rows inserted or deleted since the start of the transaction.
Serializable	Ensures that a set of transactions performed concurrently produce the same result as if they were performed sequentially.

For an up-to-date list of the relational and nonrelational databases that work with .iqd files, visit the IBM Cognos Resource Center Web site ((http://www.ibm.com/software/data/support/cognos_crc.html)).

Example - Using an .iqd File to Access a UNIX Data Source

Some of the data required for your sales-tracking model is stored in an Oracle database on a UNIX server.

You use IBM Cognos Impromptu to query the ORDERS and ORDER_DETAILS tables, saving the results in an .iqd file. You then specify this .iqd file as the source for your model in Transformer, so that you gain access to the data stored in your Oracle database.

You confirm that the server connection information in the cogtr.xml file on your modeling computer matches that on your UNIX server. The cogtr.xml file is located in the *installation_location*\configuration directory.

Transformer can now refresh the data directly from the server holding the Oracle database whenever categories are generated or cubes are created.

Modify the SQL Query in an IQD Data Source

SQL is the standard query language used to access relational database information. For Impromptu Query Definition (.iqd) source files only, expert users can modify the SQL in the **Data Source Viewer** to optimize, customize, or refine queries.

After you modify the SQL, the relationships between columns in the .iqd file and those in the model may be lost. This caution applies if your SQL contains a calculated expression, your .iqd file uses an input scale, or the column names of the items below the END SQL line do not match the column names in the SQL code.

Steps

1. From the **View** menu, click **Data Source Viewer**.

2. To open an editing window, check the **Enable Modifications** check box.

3. Click the **SQL** tab, enter your changes and click the **Preview** tab to review them.

4. Click **OK** to save your changes or click **Cancel** to close the window without saving.

Character Sets Used with Delimited Text Data Sources

Transformer is designed to handle the more common character sets used in delimited text data sources. However, consider the following when working with delimiters.

- Standard delimiters such as the comma, semicolon, or space character remain the same across character sets.

- To use the tab character as a delimiter, you must type \t (backslash t).

- Only the first byte in double-byte or multi-byte characters is examined to confirm that a character in the input stream matches the delimiter character that you specify in Transformer.

- With DOS outputs, the source data is rendered in the DOS Code Page (OEM) character set, which maps to the active DOS code page. This is typically Code Page 437, also known as the IBM PC character set, instead of the default for Windows (ANSI or Latin 1).

- Regardless of the data source, Transformer converts the delimiter character to the OEM character set.

For more information about character sets and code pages, see your Microsoft Windows documentation.

Named Ranges

You must understand named ranges before you use source data from Microsoft Excel spreadsheets, whether in crosstab or database format.

To use a crosstab as a source, you must define one or more named ranges in the spreadsheet. These ranges establish which data becomes columns in the model.

To use a database as a source, you must define a named database range in the spreadsheet, and also specify that range in Transformer. Transformer reads the named ranges from the crosstab or database, and the data class (text, numeric, or date) for each cell value.

If the **Data Sources** list contains columns identifying named ranges that you do not need, delete them from the list. Deleting columns in Transformer does not affect your spreadsheet.

Before further processing, Transformer verifies that the named range represented by each column in the model still exists in the spreadsheet and that the spreadsheet still contains the same number of pages, rows, and columns as it did when you first identified it as a data source for the model. If differences exist, you are prompted to use the **Modify Columns** command on the **Tools** menu to add, modify, match, or remove columns as required, before proceeding.

For more information about how to define named ranges, see the documentation provided with your spreadsheet software.

Example - Specifying Named Ranges for a Multipage Crosstab

This example shows one page of a multipage crosstab.

	A	B	C	D	E
1	Glassware Division	Q1	Q2	Q3	Q4
2					
3	Income-Net Sales	10,000	10,000	15,000	20,000
4	Expenses-Salary	2,000	2,000	2,500	2,500
5	Expenses-Rent	800	800	800	800
6	Expenses-Advertising	600	700	600	600
7	Expenses-Production	1,500	1,500	2,000	4,000
8	Net Income	5,100	5,000	9,100	12,000

For this crosstab, the following ranges were named, and these range names automatically become the columns in the model.

Name	Range	Name	Range
Division	A1	Expense	A4 .. A7
Quarter	B1 .. E1	Expense Amount	B4 .. E7
Income	B3 .. E3	Net Income	B8 .. E8

For a model requiring only the Division, Quarter, Expense, and Expense Amount columns, Transformer reads all the expense types and expense amounts for each quarter and for each division.

Division	Quarter	Expense	Expense Amount
Glassware	Q1	Expenses-Salary	2,000
Glassware	Q2	Expenses-Salary	2,000
Glassware	Q3	Expenses-Salary	2,500
Glassware	Q4	Expenses-Salary	2,000

Division	Quarter	Expense	Expense Amount
Glassware	Q1	Production	1,500
Glassware	Q3	Production	1,500

In your Transformer model, under Glassware, the **Data Sources** list shows the columns Expense Amount, Expense, Income, Net Income, and Quarter.

Example - Specifying a Named Range for a Spreadsheet Database

You have a simple spreadsheet database that has a single named range.

	A	B	C	D
1	EMP_NO	NAME	DEPT	SALARY
2	0256	Wilson	TECHW	50,000
3	0141	Barnes	DESIGN	60,000
4	0724	Paul	DESIGN	70,000
5	1290	Power	DESIGN	80,000

The named range for cells A1 .. D5 is DataTable.

Transformer reads rows 1 to 5 in the range DataTable. The **Data Sources** list shows the Human Resources columns EMP_NO, NAME, DEPT, and SALARY.

PowerHouse Data Source Parameters

You can specify a PowerHouse portable subfile as the data source for a model.

Dictionary component	Model equivalent
Element name	Column original name
Element heading	Column name; the PDL multiline character (^) is replaced by a space character
Item datatype - character	Data class text and size
Datatype - portable zoned	Data class: unspecified (default); Measure type: 32-bit integer

Dictionary component	Model equivalent
Datatype - portable float	Data class: unspecified (default); Measure type: 64-bit floating point
Datatype - portable date	Data class: date; date input format: predefined

Instead of using the PowerHouse dictionary values to set the **Output scale** and **Precision** properties, columns generated from a portable subfile initially have these attributes set to zero. To modify these attribute values, use the **Format** tab on the **Measure** property sheet.

Transformer automatically interprets the following components of the portable subfile dictionary.

Integer and float data from portable subfiles are represented by their .psd datatype as follows:

- Integers stored as **Portable Zoned** in the subfile appear as signed, zero-filled numbers.

  For example, the number 16, stored in **Portable Zoned Size 6** format, appears as +00016.

- Float data stored as **Portable Float** in the subfile appears in scientific notation.

  For example, the number 812,333.65, stored in **Portable Float** format, appears as +. 81233365000000000000E+06.

Using Multiple Data Sources in Your Model

You can create more powerful views of your data by adding multiple data sources to a single model, allowing you to combine disparate data sources into a single PowerCube. In Transformer version 8.x, you can model a single cube using any combination of the supported data sources, giving you a unique view of your business.

The advantages of multiple data sources include:

- measure allocation capabilities

  Measures that are not normally related to some parts of the dimensional hierarchy can have values allocated to those parts of the dimension.

  For example, suppose your sales-tracking system lacks information about staffing levels at each branch. You include a data source that provides this information so that you can derive valuable information, such as the average sales per employee.

- performance improvements

  When you split a single, large data source into smaller, denormalized sources, you can shorten the processing time if the columns in all measure-containing sources are associated with unique levels in the model. This is because Transformer can associate such measures directly, without confirming their context in the level hierarchy.

- integration of data from various sources

  Transformer does not perform database joins between like-named columns. Rather, it associates the data values in each source with the categories that are generated from the columns. To create source data that joins database tables, use IBM Cognos Impromptu, IBM Cognos 8, IBM Cognos 8 Data Manager, Composite Server, or a data access application.

Data Source Scope

Data sources may be associated with all or part of the Transformer **Dimension Map**. To make the proper associations, all column names related to the same level in the **Dimension Map** must match exactly. Also, the levels with which these source columns are associated must be either unique or fully qualified by one or more higher levels in the hierarchy.

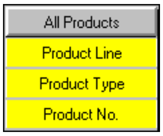
Tip: To see the scope for a data source in your model, click it in the **Data Sources** list and, from the **Edit** menu, click **Show Scope**. You can also right-click in the **Dimension Map** or the **Data Sources** list.

The scope that appears depends on the relationship between the source columns and the levels in the **Dimension Map**.

Tip: You can change the default colors of the scope map on the **Dimension Map** tab of the **Preferences** property sheet on the **File** menu. Click the color swatch you want to change and select a new color from the **Color** palette. Click **OK** twice.

Level Derived Directly

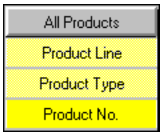
When a level is derived directly, it takes its category values from a column in the source. For example, this source contains columns that provide category values for all levels. The default color is dark yellow.



Level Derived Indirectly

When a level is derived indirectly, the source is not related to the level, but is related to descendants of the level. The default color is light yellow.

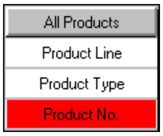
For example, this source contains a column that provides category values for the unique level Product No., but no columns for the ancestor levels. Product Type and Product Line may be levels manually created by the modeler, or their category values may come indirectly from columns in other data sources.



Level Derived From a Source with Missing Columns

When a level is derived from a source with missing columns, it cannot take its values from the source because the source is lacking columns for the ancestor levels, and a missing level, by definition, is not unique. The source does not contain enough data to map column values to a level without knowing the entire context. The default color is red.

For example, this source contains a column that provides category values for the level Product No., but not for the ancestor levels Product Line and Product Type. The context of the Product No. categories cannot be determined because the level was not declared unique.

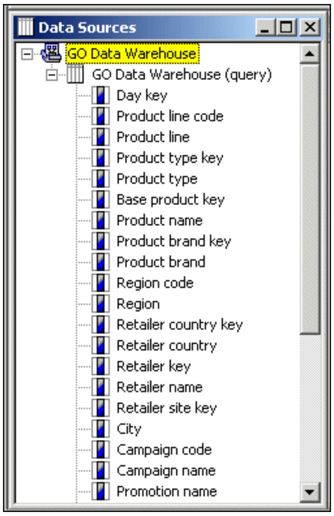


All Products
Product Line
Product Type
Product No.

Example - Verifying the Scope of Data Sources in a Sales Model

You want to verify the scope of your data sources in a basic sales model.

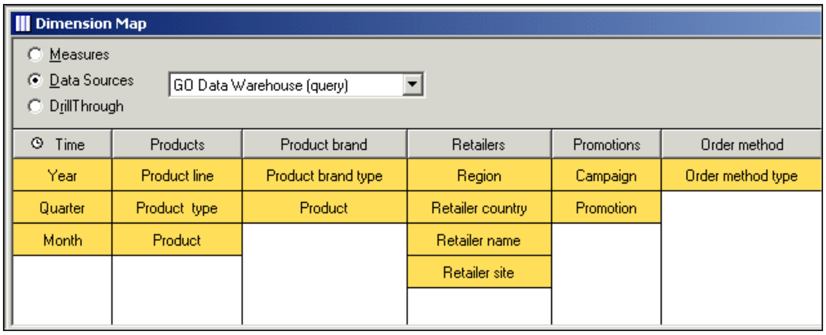
The data source is as shown. The GO Data Warehouse columns of Product Line, Product brand, and Region provide category values for the top levels in the Products, Product Brand and Retailers dimensions, and so on.



Data Sources

- GO Data Warehouse
  - GO Data Warehouse (query)
    - Day key
    - Product line code
    - Product line
    - Product type key
    - Product type
    - Base product key
    - Product name
    - Product brand key
    - Product brand
    - Region code
    - Region
    - Retailer country key
    - Retailer country
    - Retailer key
    - Retailer name
    - Retailer site key
    - City
    - Campaign code
    - Campaign name
    - Promotion name

Because GO Data Warehouse contains columns for all six dimensions, its data source scope covers all the dimensions in the model.



Dimension Map

○ Measures
● Data Sources GO Data Warehouse (query)
○ DrillThrough

Time	Products	Product brand	Retailers	Promotions	Order method
Year	Product line	Product brand type	Region	Campaign	Order method type
Quarter	Product type	Product	Retailer country	Promotion	
Month	Product		Retailer name		
			Retailer site		

Control When the Source Data Is Read

In models with multiple data sources, you can control when a data source is read. For example, a data source can be read during category generation, during cube creation, or both. You can use this capability to avoid unnecessary processing, resulting in a more efficient model design.

Here are some typical scenarios:

- After Transformer reads your purely structural data sources and populates your model with data for the required category structure, you can change the timing so that these static categories are not regenerated every time you create the PowerCube.
- For structural data sources that frequently change, you can set the timing to update the categories whenever you create the PowerCube.
- For transactional data sources, where the measure values are constantly changing, you can select the **Default** method of PowerCube creation.

Steps

1. In the **Data Sources** list, right-click the data source for which you want to control the timing and click **Properties**.
2. On the **General** tab of the **Data Source** property sheet, under the **Timing** box, select the **Generate categories** check box.

   This will cause the data source to be queried whenever categories are generated.
3. Select the **PowerCube creation** check box and choose what will occur when cubes are created or updated:
   - **Default**

     Transformer reads all columns in the source that relate to levels in the model to see if they are associated with measures. If the source is purely structural, the cube is not created or updated, and an error message appears. Use this option in a production environment for transactional data sources and structural data sources that contain non-static data.
   - **Generate categories only**

     Transformer queries only for structural information when the **Create Selected PowerCube** and **Update Selected PowerCube** commands run. If there are measures in any of the source files, they are not retrieved.
   - **Create the PowerCubes**

     Transformer queries the source and creates or updates the cubes, even if the source lacks measures. In a design and development environment, use this option to override error messages and continue processing.
4. Click **OK**.

Defining Columns

Even if you build your dimensions and measures from in-scope data source columns, you may have to change your column definitions to ensure that your model delivers solid business value.

Source columns contain not only the text, date, and code values that become the categories in your model, but also the numeric values that you select as measures, or performance indicators. Data columns can also contain values that you may want to use as alternate labels, short names, or textual descriptions for categories.

If you need more structural or numeric information in your model than is available from your source data, you may be able to add it by using calculated columns. For more information about the functions you can use when creating calculated columns, see "IBM Cognos 8 Transformer Expression Editor" (p. 379).

As the modeler, you must ensure that your model columns remain synchronized with their associated data sources. You can use the **Data Sources** list to create or delete columns, and to examine or modify column properties.

For some data sources, Transformer can automatically identify columns and assign default column names and properties. However, if your data sources include IBM Cognos 8 packages or reports, you select specific query objects from an available list.

Any time that you create columns manually, as you must for fixed-field text files, you must identify only those data columns that are used in the model. You can change these assignments later.

Troubleshooting Issues Related to Column Names

When an identically named column appears in two or more data sources, Transformer associates each column with a level that has its source column set to the common name. However, it cannot use matching columns to perform joins on the source files. Instead, it uses the common name to associate both columns with categories in the dimension map, or with a measure in the model.

For example, suppose your model has an Order Header data source and an Order Detail data source. Each contains a column named ORDER_NUMBER. Transformer cannot join an Order Header record with an Order Detail record to create a composite record describing a sales line in an invoice. However, Transformer associates the values from both of these sources with a specific level in one of the dimensions in the model.

To create a join between two columns, use a database query tool such as IBM Cognos 8 or IBM Cognos Impromptu before you create the data source.

Troubleshooting Issues Related to Date Columns

To obtain the data required to populate the time dimension for your model, you need at least one data source that includes the required date values. However, in a model with multiple data sources, date columns may appear in several source files, and date columns may not relate to your chosen time dimension.

For example, suppose your sales-analysis model contains one source file with order information, including the order date, and another that contains sales forecasts by time period. For Transformer to relate values from both sources to your chosen time dimension, you must rename the columns that contain the date values in each source file so that they use the same name.

To ensure that you associate your date values with the appropriate dimension levels and measures, without increasing the size of the model unnecessarily, we recommend that you

- Set the absolute range of dates allowed in the model.
- Match the date format in the model to the format in the source file.
- If required, specify the turn-of-the-century break-point (`CenturyBreak` setting).

- Specify whether the measure values associated with these dates apply to some or all levels in the time dimension.

If you want to take advantage of Transformer's relative time functionality, do not import the date dimensions from dimensional packages. Instead, use the **Insert Data Source** option to import the appropriate date field to create your time dimension.

Define Columns in a Fixed-field Text Data Source

When your data source is a fixed-field text file, you must define the columns using the **Column** property sheet. Otherwise, Transformer does not have the necessary information about how columns in the source file are defined and cannot accurately populate your model.

Because this is a manual process, you can define overlapping columns or define a column that includes other columns.

With fixed-field text sources, you cannot use the **Modify Columns** command on the **Tools** menu to remap model columns when the structure of the source file changes. Instead, you must manually modify the starting byte and width of each column on its property sheet.

Steps

1. Use the **New Data Source** wizard to add a fixed-field text data source.
2. From the **Edit** menu, click **Insert Column**.
3. In the **Column name** box, type a name for the new column.
4. On the **General** tab, in the **Data class** box, select the appropriate data type.
5. In the **Position** box, type the starting position of the column in a record.

   The first byte in a record is byte number 1.
6. In the **Size** box, type the width of the column in bytes.
7. Set other properties for the column as required and click **OK**.
8. Repeat steps 2 to 7 for each column in the source file.

Example - Defining Fixed-field Columns

You want to define columns for a fixed-field source file in which each record is 38 bytes long and encompasses five data fields.

12345678	90123456789012	34567890	12345678
20070103	Product1	Color1	02140330
20070103	Product1	Color2	02870335
20070103	Product2	Color1	04560508

20070103	Product2	Color3	03110388

You specify in the **New Data Source** wizard that the first field is an 8-byte numeric field giving the date of a transaction. The next 14-byte field indicates the product type. Note that the type Product1 uses fewer than the 14 bytes available in the field. An 8-byte field indicates the color of each item sold. The last 8 bytes store monetary values for the measure fields Cost (4 bytes) and Revenue (4 bytes).

Transformer can now correctly interpret the columns. For example, the first record represents a transaction that generated 330 in revenue, occurring on January 3, 2007, involving Product1, Color1, that cost 214 to produce.

Scale Input Values

You may want to change the number of decimal places or significant digits in your measures to better reflect what your users are expecting in their OLAP reports or to create calculations. Decimal values are read into the model based on a scale that you specify. This scale is the source value, multiplied by 10, raised to the power of the input scale. This scaling formula allows Transformer to handle values without integer portions, such as .0003.

To properly scale your input values, you must

- Set the **Input scale** property for the source column.
- Set the **Output scale** property for the measure that uses the source column.
- Set the **Precision** property for the measure that uses the source column.

When you create a data source using an IBM Cognos 8 package, Transformer automatically uses the scale specified in the Framework Manager query, which in turn matches the scale defined in the source database. Consequently, on the **General** tab of the **Column** property sheet, the **Input scale** option always shows a default value of zero.

Regardless of the precision supported by the source database, the overflow limit for the 64-bit floating point storage type in Transformer is 18 significant digits, excluding the decimal separator. When a measure value has more significant digits after the decimal place than is specified in the precision attribute for the model, the number is truncated and the last bit rounds up, rather than down.

Unless truncated, Transformer does not round numbers scaled using **Input scale**, **Output scale**, and **Precision**. For example, in scaling 1,792,485.86 to x.x million using an **Input scale** setting of 0, an **Output scale** setting of 6, and a **Precision** setting of 1, Transformer produces a result of 1.7 million, not the rounded-up value 1.8 million.

Steps

1. Open the **Column** property sheet for the measure that you want to scale.
2. On the **General** tab, in the **Input scale** box, type the appropriate value and click **OK**.

   Tip: For monetary amounts, this is normally the number of places after the decimal in the source data.

3. Open the **Measure** property sheet for the measure that you want to scale.

4. On the **General** tab, in the **Output scale** box, type an appropriate value.

   When this value matches the input scale specified in step 2, consistency is maintained between the source data and the measures in your reports. However, this consistency is not mandatory. Enter a value that makes sense for the reporting needs of your users.

5. In the **Precision** box, type a value for the number of decimal places to appear in the report, and click **OK**.

6. Repeat steps 1 to 5 for each source file column that needs scaling.

Example - Scaling Decimal Values to Appear as Integers

You want to scale decimal values so that they appear as integers in your reports.

You have a source file in comma-separated (.csv) format that contains the following values:

DATE,CUSTOMER,PRODUCT_LINE,PRODUCT_TYPE,PRODUCT,QUANTITY,SALE_AMOUNT

20060603,GO Outlet Montreal,Outdoor Products,Tents,40100,5,600.55

20060604,GO Outlet Montreal,Outdoor Products,Tents,40101,2,189.90

20060604,GO Outlet Montreal,Outdoor Products,Tents,40102,1,129.95

To convert these decimal values to the appropriate integer values in your Transformer model without changing the results in your data source or reports, you specify the following settings:

- an **Input scale** of 2 for the column SALE_AMOUNT

  Each value in the source is multiplied by 10 to the power of 2 (input scale), yielding an integer value. For example, 600.55 becomes 60055.

- an **Output scale** for the measure that matches the **Input scale** for the SALE_AMOUNT column

  The measure value is divided by 10 to the power of 2 (output scale) when reports are run. For example, the scaled value 60055 reverts to 600.55.

- a **Precision** property of 2 for the measure, if two decimal places are required, or 0 if the reports need not carry this level of detail

  An example is 600.55 or 600, but not 600.5500.

The result of all these transformations is that your report users see the appropriately scaled values for SALE_AMOUNT.

Set the Level of Detail for Dates

Values for some measures in a model often apply to time periods that are not at the lowest level in the time dimension. In such cases, you specify the actual level of detail to which the date values apply.

For example, actual revenue values may be derived from invoice information that accumulates on the dates that orders are filled. In contrast, sales forecasts or budgets are usually projected for months or quarters, not days. You can specify the level of detail to which forecasts and budgets apply in the time dimension for your model.

The level of detail setting that you specify for a column must be supported by the date values stored in the associated column in your data source. For example, you cannot specify a degree of detail of day if the date values are stored only as year and month data, such as 200602.

If a particular measure has meaning to only one level in the time dimension, you can allocate values for that measure to lower levels.

Steps

1. Open the property sheet for the relevant date column and click the **Time** tab.
2. In the **Degree of detail** box, select the date level appropriate to the measure in your data source.

   For example, if the source contains a measure that provides monthly forecast values, click **Month**.
3. Click **OK**.

Example - Aligning the Date Dimension with Available Data Source Measures

You want to map date dimension categories to the correct measures in a data source.

Suppose you have sales figures that are stored in the following format in your data source:

WEEK,CUSTOMER,SALES_REP,TOTAL_SALES 20060208,Fresh Air Lte 4,Francoise LeBlanc,4977.99 20060215,Fresh Air Lte 4,Francoise LeBlanc,2955.85

The date values are specified in **YMD** format, but the associated measure values are actually weekly sales summaries by sales representative.

You specify a **Degree of detail** setting of **Week** so that you report the correct values.

Specify Monthly or Quarterly Time Arrays

Your transactional data is stored as quarterly or monthly values, but in general, you roll up this information into yearly results. It may be more efficient to define the columns in your model as members of a time array, rather than as individual measures. A time array consists of four or twelve adjacent columns that contain quarterly or monthly values for one year.

You can define more than one array per model. For example, you can set up one source file with all quarterly or all monthly arrays. Or you can set up multiple source files, with one array type in each file, using a different array for each year of data. However, you must use the same format in all date columns and a different name for the first month or quarter in each array.

If you use the **New Model** wizard to create an initial model, remember to clear the **Run AutoDesign** check box. Otherwise, all the measures appear in the **Measures** list before you define your array, and you must delete them before beginning the array definition.

An array is treated as one object. Do not delete a column that is a member of an array. If you do, all other member columns in the same array are automatically deleted.

Before you import any data, ensure that your source files contain groups of contiguous columns, such as four columns for a quarterly array, or 12 columns for a monthly array.

Also, the data source or sources for the time array must contain at least one date column in addition to the columns that represent the individual elements of the array. The value in the date column must be the same for all data in the array.

To ensure that Transformer processes the array correctly, you must use the first month of your fiscal year as the date in your date column. This specifies the year-begin date for the year in which the array applies.

Steps

1. Ensure that your model does not contain any objects in the **Measures** list.
2. Double-click the first column that you want in the time array, such as Month_01.
3. On the **Column** property sheet, click the **Array** tab and, in the **Array type** box, click **4 quarters** or **12 months**.

   The **Data Class** of the subsequent columns automatically changes to **Array Member**.
4. In the **Date column** box, click the column that contains the starting month of your fiscal year.

   This is the date column, which is usually the same value as the first month in the array, in **YYYYMM** format.
5. In the **Start month** box, type the number of the month in which the fiscal year begins.
6. Click **OK**, and drag the column to the **Measures** list.
7. If you want to add another data source, click the **Data Sources** list, click the **Insert Data Source** button, and repeat step 2.
8. In the **Column name** box, type a different name to distinguish this column from the first month or quarter in the previous array. You can also enter explanatory notes on the **Description** tab.
9. Repeat steps 3 to 6 to add this new array to your model.
10. After you specify all the arrays that you want in your model and ensure that the **Measures** list contains the initial column of each array, from the **Tools** menu, click **Check Model**.
11. If there are no problems with your design, you can now create the cube and confirm your results in your OLAP reporting component.

    If there are problems with your model design, review the **Check Model** messages to identify the issues.

Example - Consolidating Quarterly Data (Single-source Model)

You want to consolidate data using a quarterly time array.

Your source data contains the total number of sales for each product by quarter. You define the Q1, Q2, Q3, and Q4 columns as members of a time array, as follows:

DATE PRODUCT Q1 Q2 Q3 Q4

200601 Product1 100 200 150 400

200601 Product2 100 175 150 350

200601 Product3 75 100 100 100

200701 Product1 110 210 160 420

200701 Product2 125 200 175 375

200701 Product3 125 150 150 150

Your OLAP reports show the following.

	2006	2007	2006+2007
Product1	850	900	1750
Product2	775	875	1650
Product3	375	575	950
TOTALS	2000	2350	4350

Observe how using time arrays yields compact and efficiently processed cubes and reports.

Example - Consolidating Monthly Data (Multiple-source Model)

You want to consolidate data from more than one source, using a monthly time array.

You have one data source for each year of data. Each source file contains the total number of items sold by retailer type (independent stores compared to department stores) by month.

You define the Month01 column in each data source as the first column in a twelve-member array. You then rename the initial column names, so that Transformer can differentiate between the two, changing the initial column in the second array (2007) to A2Month_01, to distinguish it from Month_01 in 2006.

The consolidated data for Array 1 (2006 Months) is as follows:

DATE TYPE 01 02 03 04 05 06 07 08 09 10 11 12 200601 Sports 05 06 07 05 04 03 06 04 08 02 01 09 200601 General 10 13 07 05 14 15 06 15 08 12 10 05

The consolidated data for Array 2 (2007 Months) is as follows:

DATE TYPE 01 02 03 04 05 06 07 08 09 10 11 12 200701 Sports 10 13 07 05 14 15 06 15 08 12 10 05 200701 General 20 26 14 10 28 30 12 30 16 24 20 10

In your OLAP reports, you can use nesting to show how many items were sold by each retailer type for the 12-month periods beginning 2006/01 and 2007/01. You can also show the totals sold by all retailers for each year, and for all years.

	2006	2007	2006+2007
Sports	60	120	180
General	120	240	360

	2006	2007	2006+2007
TOTALS	180	360	540

Observe how using time arrays yields compact and efficiently processed cubes and reports.

Modify Date Categories When Spanning Two Centuries

If your legacy data source represents date values using two-digit years and the data spans two centuries, you can supply a `CenturyBreak` value, or change the default break-point in the cogtr.xml file, so that your data is correctly incorporated into your model.

By default, Transformer interprets the years 00 to 19 as 2000 to 2019, and the years 20 to 99 as 1920 to 1999. For a different default setting, open the cogtr.xml file and specify a `CenturyBreak` value equal to the last two digits of the first year that you want to appear in the earlier century.

For example, set the `CenturyBreak` value to 80 so that the years 80 or higher are generated in the 20th century (1980-1999) and the years prior to 80 are generated in the 21st century (2000-2079).

If your source files have overlapping date ranges that span both centuries, such as 1900 to 1999 and 2000 to 2020, you must convert your source data to use a four-digit (YYYY) date format rather than use a `CenturyBreak` setting.

Steps

1. Open the cogtr.xml file and search for an existing `CenturyBreak` entry.

   The cogtr.xml file is located in the *installation_location*\configuration directory.

   If you cannot locate a `CenturyBreak` entry, Transformer uses the default value (20), meaning that the years 00 to 19 are interpreted as 2000 to 2019 and the years 20 to 99 are interpreted as 1920 to 1999.

2. Modify or create the `CenturyBreak` entry to match the date values in your data source by typing the following, where xx is the last two digits of the first year that you want to be in the earlier century:

   ```
   CenturyBreak=xx
   ```

3. Save the cogtr.xml file.

4. Close and restart Transformer for the change to take effect.

Create a Signon

You create a data source signon in Transformer to facilitate how Transformer accesses secured data sources when building cubes.

By defining a data source signon in the Transformer model, you can avoid having to specify the signon information that the data source requires to build the cube.

There are two types of Transformer signons: data source signons and Cognos 8 signons.

Data Source Signons

Data source signons are either imported with .iqd files, or can be created in Transformer when IBM Cognos 8 data sources have a signon defined in Content Manager that does not have a password associated with it.

Under normal circumstances, Transformer does not interact with the IBM Cognos 8 data source signon defined in Content Manager; the metadata is retrieved along with the signon and the data is returned. However, when there is more than one data source connection and each has more than one signon, and the signons are configured to prompt for a password, the user must select the appropriate signon before the data can be returned. When the data source is configured to prompt for a password, you can create a Transformer signon to enable cube builds in batch mode. For more information about building cubes in batch mode, see "Command Line Options" (p. 239).

The same ambiguity arises when there is no password associated with the signon defined in Content Manager. This configuration is used as a data source that prompts for user ID and password. When a signon has been configured without a password, you can do one of two things to build a cube in batch mode:

- Ask the system administrator to create a Content Manager signon with a valid password.
- Create a data source signon in Transformer.

  When you create a signon, Transformer assigns the signon to the package or report on which all subsequent queries are based.

  If you do not create a signon in Transformer, you will be prompted once during the session to create a signon, and again prior to running the first query based on the package or report when the saved model is reopened in Transformer.

  If you do not create a Transformer signon, you will not be able to build cubes in batch mode.

Tip: When a data source has more than one signon configured, the multiple signons are shown on the **Content Store Data Source** tab in the **Data Source** property sheet.

Cognos 8 Signons

You can configure IBM Cognos 8 to use authentication to an external namespace, where users are prompted for credentials as part of the IBM Cognos 8 logon process.

You can create Cognos 8 signons to build cubes in batch mode in this environment. This signon maintains the user ID, password, and associated namespace. Create as many Cognos 8 signons as the number of Cognos 8 namespaces to which your users need to log on. To enable Transformer to use the Cognos 8 signon automatically, enable the **Set as auto logon** property.

For more information about Cognos 8 signons, see the IBM Cognos 8 *Administration and Security Guide*.

Tip: Cognos 8 signons must have an associated namespace to be valid.

Steps to Create a Data Source Signon

1. Open the Transformer model.
2. Right-click in the **Signons** list, and click **Insert Signon**.

Tip: If the **Signons** list is not shown, from the **View** menu, click **Signons**.

3. Select the **Data source signon** check box.
4. In the **Signon name** box, type the name for the signon.
5. In the **User ID** box, type the user ID.
6. Do one of the following:
   - To prompt for a password when accessing the data source, select the **Prompt for password** check box.
   - To bypass prompting when accessing the data source, clear the **Prompt for password** check box and, in the **Password** box, type the password.
7. Click **OK**.

   The new data source signon appears in the **Signons** list and is preceded by a special icon .

Steps to Create a Cognos 8 Signon

1. Open the Transformer model.
2. Right-click in the **Signons** list, and click **Insert Signon**.
3. In the **Signon name** box, type the name for the signon.
4. In the **User ID** box, type the user ID.
5. In the **Password** box, type the password.
6. To set up an automatic logon when logging on to IBM Cognos 8, select the **Set As Auto Logon** check box.
7. In the **Namespace** list, select the appropriate namespace.
8. Click **OK**.
9. In the **Confirm Password** dialog box, type the password again and click **OK**.
10. Click **OK**.

    The new Cognos 8 signon appears in the **Signons** list and is preceded by a special icon .

Chapter 4: Structuring Your Data Into Dimensions

By structuring your data into dimensions, or hierarchies that represent major segments of your business information, you ensure that PowerCubes created from your model support your users' OLAP reporting and analysis needs.

For example, in a sales analysis model, typical dimensions include dates of sale (Time), sales locations (Regions), product and purchasing details (Products), and customer information (Customers).

You can create dimensions in Transformer manually or with the **Run AutoDesign** feature in the **New Model** wizard. When you use the **AutoDesign** tool, the time and regular dimensions are structured for you, based on patterns and relationships detected in the source data.

When you create dimensions manually, you must select the appropriate columns from your imported **Data Sources** list.

AutoDesign

AutoDesign helps you make a preliminary model design and is enabled to run by default whenever you create a new model.

Tip: If you do not want to use AutoDesign, you can either clear the Run AutoDesign check box on the last page of the New Model wizard or clear the Run AutoDesign when creating a new model check box on the AutoDesign tab of the Preferences property sheet.

When used with a supported data source, **AutoDesign** analyzes the data type, column names, and structural framework of your data file. It then automatically

- creates a time dimension based on the date column

  If the dates are not in a predefined format, Transformer may prompt you to define the format.

- adds columns with numerical values to the **Measures** list

- places all remaining columns on the **Dimension Map** using a best-fit approach

You may need to change at least some of the dimensions and measures created by **AutoDesign**. For example, it cannot distinguish numeric data representing quantities from codes with a numeric format, such as order numbers.

For a more accurate initial model, you may want to manually set the **Data class** on the property sheet for each column and specify which source columns are measures by dragging those columns from the **Data Sources** list directly to the **Measures** list.

Although **AutoDesign** is intended to help you create dimensions, source levels, and measures for a new model, you can also run it against an existing model to add dimensions and source levels from not-yet-used columns. To run the tool against an existing model, from the **Tools** menu, click **AutoDesign**.

If your data source is a fixed-field text file, you must identify the columns in the query before you run **AutoDesign**. For more information, see "Define Columns in a Fixed-field Text Data Source" (p. 74).

Create a New Dimension

You need to create a dimension for each aspect of the business that your users want to analyze.

For a relational data source, the steps are basically the same for regular and time dimensions. However, for the latter, you must specify whether to create the standard time levels (Year, Quarter, and Month) or a custom set (Lunar Week, Lunar Day, and so on).

Note: Scenario dimensions are not considered a separate dimension type. For more information about setting up scenario dimensions, see "Define a Scenario Dimension and a Cube Opening Level" (p. 95).

For relational, DMR, and OLAP data sources, you insert new dimensions from the **Dimension Map**.

Steps to Create Dimensions in the Dimension Map Using Relational Data Sources

1. In the **Dimension Map**, click a dimension in the dimension line to ensure that no level is currently selected.

2. From the **Edit** menu, click **Insert Dimension**.

   A new dimension is inserted to the right of the selected dimension.

   Tip: If the **Dimension Map** is active, but no dimension is selected, the new dimension will be inserted at the left-most position on the **Dimension Map**.

3. In the **Dimension name** box, type a name for the new dimension.

4. In the **Dimension type** box, do one of the following:

   - Click **Regular** to create any dimension that does not track time.

     For more information, see "Adding Levels and Categories to a Dimension " (p. 86).

   - Click **Time** to create a dimension that contains periods such as years, quarters, months, and days.

     For more information, see "Setting Up the Time Dimension" (p. 95).

5. When you have finished setting the required properties on each of the other tabs in the **Dimension** property sheet, click **OK**.

Steps to Create Dimensions from the Dimension Map Using OLAP and DMR Packages

1. In a Transformer model, right-click the **Dimension Map** and click **Insert Dimension from Package**.

2. Browse to select a dimensional package and click **Finish**.

3. In the dimension tree for the package, select the dimensions, hierarchies, or levels that you want to add to your model and click **OK**.

   The dimensions that you selected are added to the **Dimension Map**, and a package data source is added to the **Data Sources** list. A single query for each dimension is added under the package data source.

   Where possible, Transformer includes the relevant label and business key as the source column for each level in the imported dimension. Because different OLAP sources behave differently, you may need to change the query items used in the source or category code columns for the levels to ensure the dimension is in scope with the measures for the model.

   The import may include more metadata than expected. This allows you to refine the columns that are used within the dimension.

   Tip: To ensure conformed dimensions across different packages, the original OLAP source should use unique business keys. This will help to ensure that Transformer generates category codes that are conformed to the OLAP source from which the dimension was created.

Define a Calculated Column

A calculated column is an expression that uses other columns, functions, and constants to derive new data for the model. Use calculated columns in your dimension structure

- to create exception dimensions, or new ways of slicing and dicing your data based on a calculation from existing source columns
- to create customized date values
- to produce new measure values

When you use a calculated column as a measure, the value is derived before any rollup takes place. A calculated column is similar to a calculated measure with the following exception: if the **Regular timing** of your calculated measure is set to **Before Rollup**, no consolidation occurs. For optimal cube size and run-time performance, calculated columns are preferable to before-rollup calculated measures.

For more information about calculated measures, see "Define a Calculated Measure" (p. 123).

Note: If currency conversion is supported by your OLAP reporting component, the calculated column is initially calculated using the default (base) currency. After that, conversion occurs dynamically, followed by rollup.

Steps

1. In the **Data Sources** list, click the data source to make it active and then, from the **Edit** menu, click **Insert Column**.

2. In the **Column name** box, enter a name for the new column.

3. In the **Column type** box, click **Calculated** and click **Calculation**.

   If you have not specified a **Data class**, you are prompted to specify whether the column consists of text, date, or numeric data.

4. In the left pane of the expression editor, expand the **Columns** and **Functions** folders as needed, select each parameter you want to use, and click the right-arrow button to insert it into the **Expression definition** box in the right pane of the editor.

   Tip: You can also double-click or click and drag the parameter to add it to the calculation.

   The list of available functions varies with the **Data class**. For example, the **first-of-month** and **today** functions are only available for the **Date** data class.

   For more information, see "IBM Cognos 8 Transformer Expression Editor" (p. 379).

5. When the expression is complete, click **OK**.

Example - Using a Calculated Column to Add an Exception Dimension

You want to use a calculated column to set up an exception dimension based on a calculation from an existing source column.

From the REVENUE and COST columns, you create a calculated column MARGIN_RANGE to provide new insights into the data. You define the margin ranges for Low, Medium, and High based on the Gross Margin formula (Revenue-Cost)/Revenue, using the following if-then-else statement:

if ("Gross Margin" < 0.50) then ('Low') else (if ("Gross Margin" > 0.70) then ('High') else ('Medium'))...

Drag the MARGIN_RANGE column to the dimension line of the **Dimension Map** to form an exception dimension.

You have now segmented your data into a set of useful analytical groupings.

Example - Using a Calculated Column to Support Allocated Measures

You can add a calculated column that consists of numeric data to be used in measure allocation.

Suppose you have two sources of data about a professional sports team. The first source contains data for individual players, and the second contains current and forecast salary figures for each team.

You add a calculated column to the second data source representing each team's forecast salary based on a 13% increase for the next year, such as

"CURRENT_SALARY" * 1.13

You drag the calculated column to the **Measures** list, which makes these figures available for allocation to the player level. For more information about how to allocate measures, see "Allocating Measures" (p. 137).

Your OLAP analysis users can now see each player's salary forecast.

Adding Levels and Categories to a Dimension

Each dimension in a model contains one or more levels that represent the information hierarchies that your users can explore as drill-down levels in their OLAP reporting component. You can adjust or rename the levels to suit the organization of your data, using the Transformer category viewer (diagram).

A model may have the following level types:

- Source levels contain categories that are generated from or matched to column values in the source data. Each source level is associated with one or more columns in the source file through different association roles.
- Manual levels are drill-down levels not associated with source columns. You create and maintain the categories in a manual level, and the manual level name in the **Dimension Map** is preceded by a special icon 👤.

You can add source levels to a dimension

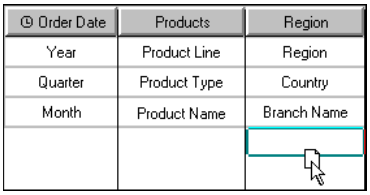
- by using the **AutoDesign** tool
- by using the **Insert Level** command in either the **Dimension Map** or the levels section of the **Categories** diagram and specifying the **Associations** for the new level
- by dragging source columns onto the dimension line of the **Dimension Map**
- by dragging source columns onto the levels section of a dimension in the **Dimension Map**

You add manual levels in the same way, but you do not specify a source column for the level.

Add Source Levels to a Dimension

Source levels obtain values for their categories from columns in the **Data Sources** list. You can add source levels to a dimension either on the **Dimension Map** or in the **Categories** diagram.

Suppose the Region dimension of your sales model has three levels: Region, Country, and Branch Name. You add sales representatives to the Region dimension by selecting the Sales Rep column from the **Data Sources** list and dragging it to the spot indicated in the following image.



🕓 Order Date	Products	Region
Year	Product Line	Region
Quarter	Product Type	Country
Month	Product Name	Branch Name

Tip: You can create a default time dimension with Year, Quarter and Month levels by setting the **Data class** on the **General** tab of the **Column** property sheet to **Date** and dragging the column to the **Dimension Map**.

Steps

1. In the **Data Sources** list, select the column for which you want to create a source level.
2. Drag the selected column to the appropriate location on either the **Dimension Map** or an existing level in the **Categories** diagram.

   On the **Dimension Map**, Transformer shows a small outlined box where it will create the new level.

   On the **Categories** diagram, Transformer inserts the new level to the left of the level on which you drop the source column.

3. Open the **Level** property sheet to set or modify the level properties and click **OK**.

   Note: Source levels can also be added by selecting an existing level on the **Dimension Map** or **Categories** diagram, clicking **Insert Level** from the **Edit** menu, and specifying the **Associations** for the new level.

Add Manual Levels to a Dimension

Manual levels provide a means of grouping categories from various source levels under a new, special category, or allow for intermediate groupings where there are too many child categories to be easily seen in the OLAP reporting components.

Because categories in manual levels typically connect to source categories in a lower level, new source categories that are not linked to a parent manual category may appear during the generation process. You can set up a temporary placeholder category in the manual level, where these new orphan categories can be placed by default.

You cannot use manual levels for subsets in your reporting component.

Tip: If you are building the same dimension in several models, you can create a spreadsheet to hold your manual categories and their associated source categories and use this spreadsheet as the data source for each model.

Steps

1. Open the **Categories** diagram for the dimension in which you want to create a manual level by selecting the dimension on the **Dimension Map** and from the **View** menu, clicking **Categories**.

2. At the top of the diagram, position the pointer over the right side of an existing level.

   The pointer changes to a crosshair.

3. Drag the crosshair to a position between two levels and release the mouse button.

   Transformer creates a new manual level and opens its property sheet.

4. In the **Level name** box, type a name for the new manual level and click **OK**.

5. For each intermediate category required in the manual level, create a category manually. For more information about manual categories, see "Create Categories Manually" (p. 89).

6. Connect the categories created in Step 5 to the appropriate child source categories in the next lowest level.

Example - Adding a Manual Level to Provide Logical Subgroups

You can add a manual level to subdivide the information for easier analysis by report users.

Suppose your staffing model contains a dimension with 14 categories at the country level. You want to subdivide the Region dimension so your users can analyze the data more easily.

You add a manual level that groups each set of countries by geographical region. You add the categories Americas, Europe, and AsiaPacific to this level. You link USA and its child categories to the Americas region. You then link the remaining country categories to their appropriate regional categories.

Create Categories Manually

You can manually add categories to either source or manual levels in any dimension by using the category viewer in the right pane of the **Categories** diagram.

If you create a category in a dimension that does not yet contain a level for it, Transformer automatically creates a new level and opens the property sheet of the new category.

Steps

1. Open the **Categories** diagram for the dimension in which you want to create a category.
2. Expand the category viewer (right pane of the diagram) as required to show the parent of the category to be added and position the pointer over the right side of the parent category.

   The pointer changes to a crosshair.
3. Drag the crosshair to the right and release the mouse button under the level that is to contain the category.

   Transformer creates a new category and opens its property sheet.
4. In the **Category code** box, type a name that uniquely identifies the category in the dimension.
5. If the category is in a source level, enter the name of the associated source column in the **Source value** box.

   For the category in the model to be matched accurately, the category name must be unique among all the source categories owned by the parent.
6. Set other properties for the category as required and click **OK**.

Create Calculated Categories

You can use calculated categories to add commonly requested calculations to your model, such as month-by-month percent growth or market share. Because the calculations are computed in Transformer and then added to the PowerCube, the results immediately appear in the OLAP reporting component for every measure specified in the cube.

For example, you can base a calculation on the special time categories **Current Month** and **Last Month**. You use the **percent-growth** function to create a new calculated category named Monthly Growth which shows the percentage change between these two items for all measures included in your PowerCube.

Category calculation is based on a formula and a set of categories to which the formula applies. You compose the formula with the help of an expression editor and a selection list of functions and operators. You then specify the categories to which the formula applies: either a single category, a category set (if supported for that function), or all categories in the level. Sets are convenient category groupings that may be from the same level, or from different levels.

You can create calculated categories at the dimension level by using the **Dimension** property sheet, or individually at the child category level, by using the **Change to Calculated Category** command on the **Diagram** menu. When created at the dimension level, new calculated categories become siblings of the categories from which they are derived.

You can view a category calculation by opening the **Calculated Category** property sheet or by using the **Show Category Calculation** command on the **Diagram** menu.

The formula varies according to the context of each category. The applicable formula can only reference another single category, not a category set or a level. Separately defined categories are useful when you want to create special groupings, such as `'Japan'+'Hong Kong'`. Calculated categories are identified in the **Categories** diagram by their own icon ▦.

Steps to a Create Calculated Category at the Dimension Level

1. Open the property sheet for the dimension that is to contain the calculated category and click the **Calculation** tab.

2. Click **Add**.

3. In the **Label** box, enter a name for your calculated category.

4. If you want all the calculated categories listed together in your OLAP reporting component, select the **Group calculated categories together** check box.

5. Click **Calculation**.

6. In the left pane of the expression editor, expand the **Functions** folder, select the mathematical function you want to use, and click the right-arrow button to insert it into the **Expression definition** box in the right pane of the editor.

   Tip: You can also double-click or click and drag the parameter to add it to the calculation.

7. Next, expand the **Levels** folder and select the appropriate level for the expression.

   Tip: You can also select and drag the appropriate level from the **Categories diagrammer**, or lower pane of the **Categories** diagram, directly into your expression, and type or copy-and-paste a valid calculation. For more information, see "IBM Cognos 8 Transformer Expression Editor" (p. 379).

8. Click **OK** in the expression editor when you are ready to save your final expression.

Steps to Create a Single Calculated Category

1. Create a manual category in the level where you want the calculated category.

   For more information, see "Create Categories Manually" (p. 89).

2. Select the manual category, and from the **Diagram** menu, click **Change to Calculated Category**.

3. In the left pane of the expression editor, expand the **Functions** folder, select the mathematical function you want to use, and click the right-arrow button to insert it into the **Expression definition** box in the right pane of the editor.

   Tip: You can also double-click or click and drag the parameter to add it to the calculation.

4. Next, double-click **Single Category** and observe that a **Category code** drop location appears in the right pane of the expression editor.

   The number of drop locations corresponds to the number of parameters in the selected function.

5. Select and drag individual categories from the **Categories diagrammer**, or lower pane of the **Categories** diagram, to each Category code drop location in the **Expression definition** box.

   Tip: You can also select and drag the appropriate level from the **Categories diagrammer** directly into your expression, and type or copy-and-paste a valid calculation. For more information, see "IBM Cognos 8 Transformer Expression Editor" (p. 379).

6. Click **OK** in the expression editor when you are ready to save your final expression.

Steps to Define Category Sets for Calculated Categories

1. If you have not already done so, generate categories for your model, and open the property sheet for the dimension in which you want to define a set.
2. Click the **Calculation** tab and click **Add**.
3. In the **Label** box, enter a name for your calculated category.
4. If you want all the calculated categories listed together in your OLAP reporting component, select the **Group calculated categories together** check box.
5. Click **Calculation.**
6. In the left pane of the expression editor, expand the **Functions** folder, select the mathematical function you want to apply to your set, for example **share**, and click the right-arrow button to insert the function into the **Expression definition** box in the right pane of the editor.
7. Double-click **Category Set** and observe that **Set 1** appears in both the left and right panes of the expression editor.
8. Expand the levels in the **Categories diagrammer**, so you can select categories for the set.
9. Select and drag the categories in your set, one by one, to the drop location for **Set 1**.
10. Double-click **Single Category** or **Category Set** again, as appropriate for your calculation, and drag the required category or categories to the drop location in the right or left pane, respectively.
11. When you have defined all your required sets and have selected, copied, or typed the rest of your calculated expression in the right pane, click **OK**.
12. Click **Generate Categories** and, if you previously closed the **Categories** diagram, reopen it by clicking **Show Diagram.**
13. Expand the levels to show your new calculated categories and verify that they appear correctly in the **Categories** diagram.

Order Categories Within Levels

By default, categories appear in the category viewer in the order encountered during query processing. When a new category is generated, Transformer places it at the end of the child category list for its parent category. You can sort the categories alphabetically or numerically, in ascending or descending order, based on values in the source column or another column.

You can order individual categories by dragging them from one spot in the **Categories** diagram to another. However, the recommended method is to specify an **Order by** column for the level that contains the categories and then modify the **Order value** property for individual categories in that level. The order values will then be automatically applied whenever new categories are added to the model.

For example, suppose the categories in the Product Type level appear in the same order as in the source file, but you want them to appear in alphabetical order. You select Product Type as the **Order by** column, and specify that categories be sorted in ascending order.

If you create data sources using IBM Cognos 8 reports, any sorting or grouping defined in the reports is not supported. Similarly, if you set the **Auto Group & Summarize** query property in a Report Studio report, Transformer does not support the resulting report groupings.

Steps

1. Open the property sheet for the level containing the categories to be ordered and click the **Order By** tab.

2. If the level is a convergence level accessible from two or more drill-down paths, then, from the **Drill-down** box, select the drill-down path in which the categories are to be ordered.

   Note: If the level is only a member of one drill-down path, the **Drill-down** box is not shown.

3. In the **Sort-by column** box, click **Add** to specify the column whose values will be used to determine the sort order.

4. In the **New Association** dialog box, click **More** to select the sort-by column, and then click **OK** twice.

5. In the **Sort order** box, click either **Ascending** or **Descending**.

6. In the **Sort as** box, click either **Alphabetic** or **Numeric** to specify whether values are to be interpreted as text or numbers during the sort.

7. Click **OK**.

Order Categories Using a Global Preference Setting

You can specify that all categories in the model use a particular sort order, rather than manually setting the order-by preference for each category.

You can still specify a different **Order by** association for a given level/drill combination. However, if you do not, the category label is used as the sort value, and ascending is always the order used.

Steps

1. Open the model and, from the **File** menu, click **Model Properties**.

2. On the **General** tab, select **Use the preference setting** in the **Default category ordering** box to order categories globally, and click **OK**.

3. From the **File** menu, click **Preferences** and click the **General** tab.

4. Select the **Order categories by default** check box and click **OK**.

Create Unbalanced Level Hierarchies Within a Dimension

You can use subdimensions to provide different levels of detail for specific categories, also known as unbalanced hierarchies.

For example, some branch offices may report product sales down to the item level, whereas others may report only to the product level. You can create a subdimension for those branches that report to the item level.

Similarly, your time dimension may contain levels for year, quarter, and month, but your OLAP report users may not need to see the month values for the previous year. You can use a subdimension to retain month levels in the current year while removing them from the previous year.

Note: If a level contains a subdimension, the name of the level appears with an ellipsis next to it on the **Dimension Map**.

Categories in a subdimension are independent of levels in other parts of the dimension. Changes made to levels outside the subdimension do not affect categories in a subdimension, and vice versa.

Steps

1. Open the **Categories** diagram and select the category below which you want to create a subdimension.

2. From the **Diagram** menu, click **Create/Delete Subdimension**.

   When you expand the selected category in the **Categories** diagram, you see a box enclosing that category and all categories below it.

3. Modify the categories in and outside the subdimension, as required.

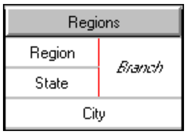
4. Repeat for as many subdimensions as your users need for their business analyses.

Drill-down Paths

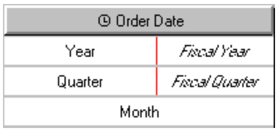
Organizing the data in your model into meaningful hierarchies enables your OLAP report users to analyze the business information at various levels of detail. Each dimension consists of one or more drill-down paths that typically contain several drill-down levels.

For example, a typical time dimension consists of the years, quarters, months, weeks, and days when sales were made. The Products dimension organizes your sales items by type, brand, model, color, and packaging. The Regions dimension allows users to drill down to the data by two distinct paths.

As illustrated here, the primary path includes Region, State, and City levels, whereas the alternate drill-down path converges on the City level by means of a Branch level.



You can also set up paths that meet at a shared convergence level in the time dimension, such as Year and Quarter, and Fiscal Year and Fiscal Quarter levels. Both drill-down paths converge on the Month level. That way, your OLAP report users can drill down to the monthly data by either the calendar year or their fiscal year paths.



Order Date	
Year	*Fiscal Year*
Quarter	*Fiscal Quarter*
Month	

Tip: You can view and manipulate drill-down paths from the **Dimensions** pane of the **Categories** diagram.

Create an Alternate Drill-down Path

You set up an alternate drill-down structure in a dimension to provide a different perspective on the data. Each alternate path connects to the primary path at the convergence level.

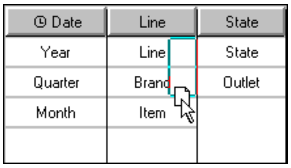
Note: In IBM Cognos 8, alternate drill-down paths are also referred to as alternate hierarchies within the same dimension.

When you connect several parent categories to the same convergence category, you must ensure that each category in the convergence level is unique and unambiguous. No two categories in the level can be derived from the same source value. Transformer prompts you to confirm uniqueness when you create an alternate drill-down path. For more information about resolving uniqueness problems, consult the index.

Because the category values at the convergence level and below are shared by all drill-down paths, removing or changing a category in one path at or below the convergence level immediately affects the same category in all other drill-down paths.

Steps

1. In the **Dimension Map**, select the level that is to be the convergence level for the new alternate drill-down path and, from the **Edit** menu, click **Create Drill-Down**. Alternatively, drag a column to the **Dimension Map**, as shown in the following image.



Date	Line	State
Year	Line	State
Quarter	Brand	Outlet
Month	Item	

Note: For a level to be a convergence level, it must be designated **Unique** on its property sheet and the category values in the level must have unique and unambiguous source values.

2. If appropriate, add intermediate (manual) levels to the new drill-down path. For more information, see "Add Manual Levels to a Dimension" (p. 88).

3. To ensure you have not introduced uniqueness problems, from the **Tools** menu, click **Check Model**.

   For more information about validating a model, see "Verify Your Model" (p. 117).

Define a Scenario Dimension and a Cube Opening Level

Scenario dimensions are often used when budgeting and forecasting, or creating PowerCubes for planning-related applications.

Although scenario dimensions are not a distinct dimension type, you can flag any dimension other than the time dimension so that its data never rolls up to either the root category or a designated parent category, which remains hidden.

You first designate a level that is a child of the non-selectable upper level in your scenario dimension as your new default opening level. You then set the **Hide the value** option for the root or parent category, to ensure that values shown at your chosen default level never roll up to this higher level.

We recommend that you always set a default opening level for your PowerCubes. You thereby ensure that cubes containing budget values or other scenario-like data do not display zeros, N/A, or meaningless numbers when opened by your report users.

When you specify a default category that is not visible to some users because of their security profile, the default category actually shown is the highest one in the hierarchy that the users are authorized to see. The default view for those users will show **na** for all measures if the default category has the **Hide the value** option enabled.

Steps

1. Open the **Categories** diagram view of your model, right-click the category whose values you want to show when the cube opens, and click **Set as Default Category**.
2. Right-click either the root or the parent category for your scenario categories, click **Properties**, and on the **General** tab, select the **Hide the value** check box.
3. Repeat Step 2 for each scenario dimension.

   Note: There is no limit on the number of scenario dimensions you can define in a cube. As long as it is not a time dimension or a measure hierarchy, any cube dimension can be handled in this way.
4. From the **Run** menu, click **Create PowerCubes** to build the cube, and open it in your OLAP reporting component.
5. Confirm that expected category values appear, rather than the values for the non-selectable root or parent category, for each dimension identified as a scenario dimension.

Setting Up the Time Dimension

The time dimension in your model contains time categories that are meaningful to your OLAP report users, such as financial accounting periods or the dates of sales transactions. The following time periods are supported:

- conventional date periods, such as years, quarters, months, weeks, and days
- industry-specific periods, such as 13-week manufacturing periods
- custom periods, such as fiscal years, hours, or minutes

- lunar time periods, such as lunar years or months
- relative time periods, such as year-to-date or previous quarter

Some time properties are always true. For example, there are always twelve months in a calendar year, and four weeks in a lunar month. Because these standard properties have been programmed into Transformer, you can work with time in ways not possible with other dimensions. For example, you can set up relative time categories to track period-by-period changes in the measures in your model.

On the dimension line, a time dimension is identified by this icon 🕒.

Time dimensions contain date levels arranged in descending order. The date levels are usually some combination of Year, Quarter, Month, Week, or Day. Transformer generates categories for the levels in a time dimension by applying date functions to the source column that you associate with the time dimension.

Transformer prevents you from creating more than one time dimension in a cube that has a time-state rollup applied. Even if this is not the case, we recommend that you create only one time dimension per cube, to avoid confusion. If you need to track both calendar and fiscal year results, set up alternate drill-down paths in a single time dimension, converging at a common level such as month. Or, if you want to compare values from two date columns, such as the elapsed time between the Order Date and Shipping Date, use a calculated column rather than two time dimensions. For more information, see "Track Monthly Performance Measures in Different Time Periods" (p. 111).

You are required to specify both the date source column and the **Dimension name** for your time dimension. You can insert manual levels into a time dimension, but you must specify a date function for each level so Transformer knows how to relate the categories to their parent levels during category generation.

For example, suppose you insert the manual level Half Year between the source levels Quarter and Year. After adding the two required categories to your new level, you must also remember to connect Q1 and Q2 to the first half, and Q3 and Q4 to the second half. Otherwise your time categories will not generate properly.

If the standard date functions do not meet your users' needs, you can manually create a custom time dimension. For example, you can add irregular work shifts or handle time-related data that comes from more than one source column. You can mix levels that use date functions with levels that derive their categories entirely from other source columns, such as Sales Promotions for a time period. However, these non-date levels will not be generated if you use the **Generate Date Categories** command on the **Run** menu.

In Transformer 8.x, you cannot import a time dimension from another OLAP data source. You must use a date field as the source for your time dimension instead.

Creating the Time Dimension

To create a time dimension, you can use any of the following methods:

- Use the **Date Wizard**.

  Transformer prompts you for information about the time dimension and then creates the dimension for you.

- Manually create a new time dimension and then successively drag the required date columns to each level in the dimension.

  Set the appropriate date function and other date-related properties for each level.

- Drag the time dimension source category from the **Data Sources** list to the **Dimension Map**.

  Note: The **Data class** of the time dimension source category must be set to **Date**.

  This automatically creates a time dimension with the standard levels Year, Quarter, and Month

- Design the dimension before you have any items in the **Data Sources** list.

  Insert a new dimension, setting the **Dimension type** to **Time**. Click the **Time** tab on the **Dimension** property sheet, and you are prompted to specify the source column containing the dates for your new time dimension in the **Date Level Creation** dialog box. Click **OK** twice, and from the **Run** menu, click **Generate Date Categories**.

  Note: If you manually type the name for the source column in the **Date Level Creation** dialog box, you are warned that the columns do not exist in the **Data Sources** list.

  Transformer uses standard date functions to generate categories in the levels of the time dimension without actually referring to a data source. Later, when the source file is available, you can add it to the **Data Sources** list and regenerate the time dimension categories by using data from the source column. The name you specify for the levels must match the source column name.

Whichever method you choose, you must define or confirm the date input format. You may discover that some of your data sources include information about their columns (sometimes called metadata), while others do not. Transformer requires information about how dates are formatted in order to correctly interpret them.

Most date formats can be automatically determined during the **AutoDesign** process. In the time dimension, if Transformer generates a category named **Invalid Dates**, it is likely that the date format is not defined, or not properly defined, for the values in the source file. For flat files such as .asc or .csv files, the date format is predefined in the source file. To change this default setting, you must open the property sheet for the column that contains your dates, click the **Time** tab, and use the **Date input format** box to specify the format you want to use.

Create a Time Dimension Using the Date Wizard

You can use the **Date Wizard** to speed up creation of a time dimension for your model. Whether your time dimension contains standard calendar or lunar time periods, down to the Month, Week, or Day level, this wizard prompts you for the information required in a logical sequence.

Later, if you decide to change the information, you can do so manually.

Steps

1. From the **Tools** menu, click **Date Wizard**.

2. Type a name for the new time dimension and click **Next**.

3. Choose the source column that contains the date values for the new dimension and click **Next**.

4. Respond to the remaining prompts.

   You can click **Back** to return and change your response to a previous prompt.

5. When you have defined the levels in the time dimension, click **Finish**.

Create a Time Dimension Manually

You can create your time dimension manually to meet the particular needs of your users. For example, you may want to provide additional relative time categories to show period-by-period changes in the measures of your cube.

You can manually add extra date levels below or between those supplied by your data source. You can also set up alternate drill-down paths in your time dimension, as long as the ancestor levels for each path are compatible.

For example, both **Calendar year** and **Calendar quarter** are valid parents of **Calendar month**. Similarly, both **Lunar year** and **Lunar quarter** are valid parents of **Lunar month**. Additional valid time periods for each quarter include months with a **4-4-5 week pattern**, **4-5-4 week pattern**, and **5-4-4 week pattern.** The convergent level for your drill-down paths, such as Week or Day, can have either calendar or lunar parent levels.

Steps

1. From the **Edit** menu, click **Insert Dimension.**

2. In the **Dimension name** box, type a name for the dimension.

3. In the **Dimension type** box, select **Time.**

4. Click the **Time** tab.

5. In the **Date Level Creation** dialog box, select **Create standard levels** (**Year/Quarter/Month**) to allow Transformer to automatically specify the required date structure.

6. In the source column list, select the column that contains values for the dates in the dimension and click **OK** twice.

   Transformer creates the new time dimension, adding the standard levels Year, Quarter, and Month.

7. If you want to add a new level to the manual time dimension, drag the source column for the time dimension from the **Data Sources** list to the new time dimension.

8. Open the property sheet for the new level.

9. In the **Level name** box, type a name that reflects the date function you plan to apply to this level, such as **Week.**

10. Click the **Time** tab.

11. In the **Date function** box, select the function to apply to this level, such as **Week** and click **OK.**

You can now change any of the following properties, which are usually set automatically when you create the time dimension for your model:

- the **Associations** for the time dimension

  Set this property on the **Source** tab of the **Level** property sheets.

- whether to always include all date categories, or only those for which source values exist, when calculating relative time categories

  Set this property in the **Inclusion** box on the **General** tab of the **Level** and **Category** property sheets.

- the date function for each time level in the model, such as **Year**, **Quarter**, and **Month**

  Set this property on the **Time** tab of the **Level** property sheet.

- the format used for date values shown in the category viewer

  Set this property on the **Time** tab of the **Level** property sheet.

- the column used to order the categories in each date level

  Set this property on the **Order By** tab of the **Level** property sheet.

Format Date Values

If you prefer a date format that differs from the predefined default, you can change the format used in the **Categories** diagram and resulting OLAP reports.

Steps

1. Open the **Level** property sheet for the date level whose format you want to change.

2. On the **Time** tab, click **Modify Format.**

3. If the format you want is listed in the **Format codes** box, select it and click **OK.**

4. To create a new format, edit the entry in the **Code** box and click **OK.**

   For information about supported date codes, see "Date Formats and Functions" (p. 342).

Set up Fiscal Years, Quarters, and Months

By default, dates in the time dimension are organized in accordance with the standard calendar year, with a **Year begins** property controlling the date when the year starts. To set up a time dimension for a non-calendar fiscal year, you change the **Year begins** property from January 1 to the first day of your fiscal year, as applicable.

If you are creating a fiscal year as an alternate drill-down path in an existing time dimension based on the calendar year, ensure that the two paths converge at a level whose categories coincide exactly.

Steps

1. Open the **Categories** diagram for the time dimension that is to be based on fiscal years.

2. Open the **Drill Category** property sheet for the drill category and click the **Time** tab.

3. In the **Year begins** box, type the date on which the fiscal year begins.

Choose a valid date from any year, but ensure that your specified starting date is the first day in the first week of that year. The default format is `YYYYMMDD`.

4. Click **OK**.

Set up Calendar and Fiscal Years Within a Single Time Dimension

In many businesses, measures are tracked over more than one time scale. A common combination is calendar and fiscal years, where the fiscal year spans different parts of two calendar years.

Typically, calendar and fiscal years span different months, converging at the month level, with the same number of months in each drill-down path.

When you create alternate drill-down paths for your calendar and fiscal time periods, the **Order by** column for each path must be identical at the convergence level. Transformer automatically handles this for the first drill-down path. However, if you add more than one alternate path, you must specify the appropriate **Order by** column for each new path.

Steps

1. If no time dimension currently exists, create one that contains standard calendar time periods, either manually or by using the **Date Wizard**.

2. In the **Dimension Map**, select the level at which you want the calendar and fiscal years to converge.

   Note: When connecting alternate drill-down paths in the time dimension, the **Year begins** property on the **Time** tab of the **Drill Category** property sheet for an alternate path must be offset by whole units of the chosen convergence level.

3. From the **Edit** menu, click **Create Drill-Down**.

   An alternate drill-down path appears in the time dimension.

4. In the **Dimension Map**, click on the empty area created for the new alternate drill-down path.

5. From the **Edit** menu, click **Insert Level**.

6. In the **Level name** box, type a name for the parent of the convergence level in the new drill-down path.

   For a fiscal year path connected at the Month level, the parent level name is Fiscal Quarter.

7. In the **Associations** box, click **Add**.

8. In the **New Association** dialog box, select **Source** from the **Association role** drop-down list, and click **More** to select the column that contains date values for the dimension. Click **OK** twice.

9. Click the **Time** tab.

10. In the **Date function** list, select the function to apply to the level and click **OK**.

11. To add additional levels to the alternate drill-down path, select the level you just added and follow steps 5 through 10 for each new level.

12. Open the **Categories** diagram for the time dimension.

13. Open the property sheet for the drill category of the alternate drill-down path and click the **Time** tab.

14. In the **Year begins** box, type the date on which the fiscal year starts and click **OK**.

    For example, if the current fiscal year started on April 1 of 2006, type 20060401.

Specify How Weeks Split When Spanning a Higher-level Time Period

In a time dimension that includes weeks, you can choose from the following options to handle weeks that span a higher-level time period:

- Always split the spanning week into two separate weeks, each of which includes the days that occur in the higher-level time period.

  Each part of the split week appears in the **Categories** diagram as a separate week. For example, the week beginning Sunday, December 31, 2006 and ending Saturday, January 6, 2007 appears as two weeks: 20061231 (a child of December) and 20070101 (a child of January). **Always Split** is the default setting.

- Split the spanning week into two distinct weeks, but not if a 1-day week is created as a result. Select the **Split > 1 day** setting.

  For example, Sunday, December 31, 2006 appears in the same week as 20070101.

- Place the spanning week in a specific time period: **First Period**, **Last Period**, or **Largest Period**.

Steps

1. Open the **Categories** diagram for the time dimension.

2. Open the **Drill Category** property sheet.

   If the dimension contains multiple drill-down paths, open the **Drill Category** property sheet for the drill-down path you want to change.

3. Click the **Time** tab.

4. From the **Partial weeks** box, select the week-spanning rule to apply and click **OK**.

Example - Changing How Partial Weeks are Handled at the End of the Year

In this example, you change how partial weeks are handled at the end of the calendar year.

Because your business aligns its fiscal year with the calendar year, you do not want Transformer to apply the default setting **Always Split** when reporting the data for partial weeks at the end of the calendar year.

In the **Partial weeks** box, you select the **First Period** setting. This places the spanning week into the year in which that week begins, so that the data for that week is accorded to December, the last month in the first fiscal year.

The days in the partial week are associated with the December 2006 time period, as follows:

Sunday	Monday	Tuesday	Wednesday	Thursday	Friday	Saturday
31	1	2	3	4	5	6

Set up Lunar Time Periods

Many businesses track performance according to the lunar calendar: that is, a year has 52 weeks, with seven days in each week, yielding 364 days.

Lunar time dimensions may contain lunar years, lunar quarters, lunar months, and other reporting periods such as 4-4-5 week months, 4-5-4 week months, and 5-4-4 week months.

Whereas lunar quarters or months must be placed in a lunar time hierarchy, you can add weeks and day levels to your hierarchy, as children of either lunar or standard calendar time periods.

To display the ending year (rather than the starting year) as the label for a lunar fiscal year that spans two calendar years, select the **Enable lunar fiscal labelling** option on the **General** tab of the **Preferences** property sheet on the **File** menu.

On the **Categories** diagram, the default label for each lunar month uses the format `YYYYMM`. The lunar months are numbered in sequence, beginning each year with the value 01, as specified in the **Year begins** property.

Lunar weeks and lunar days are labelled in`YYYYMMDD` format, where `DD` is the first day of the week on the standard calendar.

You can use the **Date Wizard** to create lunar time dimensions, and choose your required lunar time periods when prompted.

Because lunar years comprise 52 weeks and not 365 days you must ensure that the **Year begins** and **Week begins on** properties coincide. You have two options:

- You can reset the **Week begins on** property for each year.
- You can accumulate the remainder days in a new category using the **Add an extra week** setting on the **Drill Category** property sheet.

  We recommend choosing this method if you want to keep the lunar year aligned with the calendar year or your fiscal year, a requirement in most business operations.

Steps

1. On the **Dimension Map**, create a new time dimension with automatically added levels.
2. Open the property sheet for the Year level and click the **Time** tab.
3. In the **Date function** box, select the **Lunar year** function to create a level based on lunar years.
4. Repeat steps 2 and 3 for the Quarter and Month levels, selecting the appropriate function.

Specify How Extra Weeks Get Added to Lunar Years

A lunar year contains 52 weeks of seven days each, for a total of 364 days. This represents either one or two fewer days than the standard calendar or leap year, respectively. However, your model design must keep each lunar year aligned with its specified **Start-of-Year** day, while not falling too far out of alignment with the calendar year.

To support this goal, you can specify whether the extra one or two days get added as an extra week in the last month or the last quarter of the year.

When you create alternate drill-down paths in a lunar time dimension, if the convergence level is **Week**, **Lunar month**, or **Lunar quarter**, you must match both the **Week begins on** setting and the **Add an extra week** setting for all drill-down paths.

If the last lunar month in the year has five weeks in a 4-4-5 week pattern, the surplus days create an extra week. The extra week is added to the previous lunar month to make a 4-5-5 week pattern, rather than a 4-4-6 week pattern, which is not valid.

Steps

1. Open the **Categories** diagram for the lunar time dimension you want to modify.
2. Open the property sheet for the drill category that you want to change and click the **Time** tab.
3. In the **Add an extra week** box, select the setting to apply to the extra days of each year.

Limit the Range of Dates Included in the Model

When you create the time dimension, you can limit the range of acceptable dates so that categories that are irrelevant to your users do not appear in their OLAP reports.

When Transformer encounters date values outside your specified range, it generates an **Early Dates** or a **Late Dates** category, or both, depending on when the out-of-range dates occur. In addition, if there are dates that are neither early nor late, but can not be placed within the specified range, Transformer generates an **Invalid Dates** category.

Steps

1. Open the property sheet for the time dimension whose absolute range you want to change and click the **Time** tab.
2. In the **Earliest date** and **Latest date** boxes, type the dates that represent the lower and upper boundaries of the range respectively and click **OK**.
3. On the dimension line of the **Dimension Map**, click the time dimension and, from the **Run** menu, click **Generate Date Categories**.

   By default, the **Range** box shows the date settings you specified in step 2. However, you can change these if you want to generate a different range of date categories.
4. Open the **Categories** diagram and check for an **Invalid Dates** category. If this is present, repeat steps 1 to 3, making any necessary corrections.

Set up a Custom Time Dimension

Instead of using standard date levels, with their built-in definitions and functions for calendar and lunar time periods, you can create a time dimension that tracks measures over custom time periods, such as work shifts and hours within shifts, or project phases and timed sub-phases.

For example, suppose you need to track the temperatures of various pieces of equipment in an electric generating station. The raw data is captured every two hours, and stored in a database. The first few rows are as follows:

Shift	Hour	Plant	Equipment	Temperature
01	0200	01-6EL	Primary Boiler	235
01	0400	01-6EL	Primary Boiler	237
01	0600	01-6EL	Primary Boiler	233
01	0800	01-6EL	Primary Boiler	235
02	1000	01-6EL	Primary Boiler	228
02	1200	01-6EL	Primary Boiler	232
02	1400	01-6EL	Primary Boiler	231
02	1600	01-6EL	Primary Boiler	233

You design your model to track the data at the Hour and Shift levels, where each work day consists of three 8-hour shifts. From this model, you create a PowerCube that is incrementally updated every eight hours. This provides your maintenance crews with the critical readings they need in a timely manner.

After you have created a custom time dimension, you can set up relative time categories for the periods in that dimension. For example, if your model is designed to show the number of patients monitored during each hour of a nursing shift, it can also show the number monitored in the same hour of the previous shift. For more information, see "Setting Up Relative Time Categories" (p. 105).

Steps

1. Click anywhere on the **Dimension Map** and, from the **Edit** menu, click **Insert Dimension** to add a new dimension.
2. In the **Dimension name** box, type a meaningful name for the new dimension.
3. In the **Dimension type** box, select the **Time** option.
4. Click the **Time** tab.
5. In the **Date Level Creation** dialog box, select **Do not create levels** and click **OK** twice.

6. Position the cursor in the **Dimension Map**, below your new time dimension and, from the **Edit** menu, click **Insert Level** to open the property sheet for your first manual level.
7. In the **Associations** box, click **Add**.
8. In the **New Association** dialog box, select **Source** from the **Association role** drop-down list, and click **More** to select the column that contains the date values for this first level of your time dimension. Click **OK** twice.
9. While still on the **Level** property sheet, assign other required properties for this level.

   For example, specify a different name for the level and, on the **Time** tab, change **Date function** or **Time level ranking**.
10. When you are finished defining the properties for this level, click **OK**.
11. Repeat steps 6 through 10 for each custom time level you want to add.

Setting Up Relative Time Categories

Some of the most commonly requested reports in any organization are period-over-period performance reports and trend analyses, how sales in the current period compare to sales from previous periods, and how last year's budget compares to projections for next year. By including the most commonly required relative time categories in your cubes, you avoid the need to recalculate them in every OLAP report.

Relative Time Categories

Models can contain three different types of relative time categories:

- single-category periods, such as **Same Month, Prior Quarter** or **Same Month, Prior Year**
- to-date periods, such as **Year To-Date** or **Quarter To-Date**
- N-period running totals, such as a 2-week total in the previous month, or a 4-month total in the previous year

You can also create custom to-date and N-period relative time categories to span specified time ranges.

When using relative time categories, ensure that you set the **Inclusion** property to **Always include** for each level in the time dimension so that the relative dates are calculated correctly.

Default Relative Time Categories

The following relative time categories are automatically inserted into your model. However, you can delete them or replace them with custom time categories that better meet your users' OLAP reporting needs.

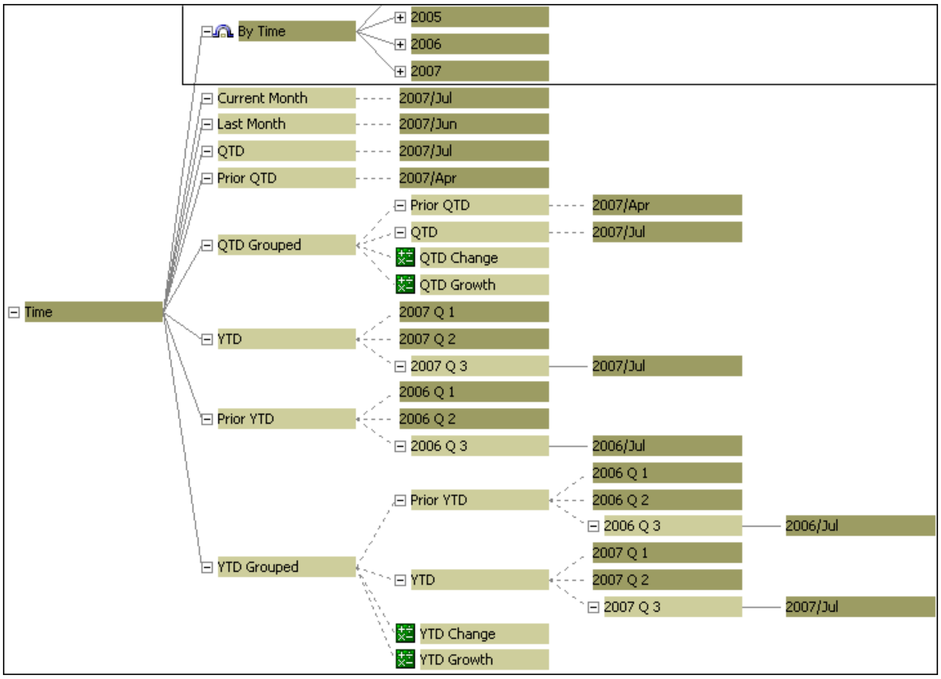
Tip: To replace automatically generated relative time categories with custom time categories, start with the closest built-in choice, change it to a custom relative time category and then change only those few settings needed to customize the category.

- Current <period>

- Last <period>
- <Higher-level periods> To-Date (Grouped), including
  - <Higher-level periods> To-Date
  - Prior <Higher-level periods> To-Date
  - <Higher-level periods> To-Date Change (automatically calculated categories)
  - <Higher-level periods> To-Date Growth (automatically calculated categories)

The periods depend on the date levels defined in your model.

In the Transformer **Categories** diagram, relative time periods appear in the category viewer as special categories, highlighted in pale green, below the other drill-down paths in the time dimension. Grouped special categories only appear in reports when the time dimension contains one or more levels higher than the current period.



Set the Current Period Automatically

Although the current period is set automatically, you should explicitly select the column to use from the **Data Sources** list. If two or more data sources contain columns that provide date values, you will avoid ambiguity by explicitly clearing the **Sets the current period** check box in the other **Data Source** property sheets.

Steps

1. In the **Data Sources** list, successively open the property sheet for each data source containing a date column for the time dimension.
2. On the **General** tab, clear the **Sets the current period** check box for all but the one you want used to set the current period.

3. Open the property sheet for the time dimension and, on the **Time** tab, select the **Automatically set the current time period** check box. Click **OK**.

Set the Current Period Manually

If you clear the **Automatically set the current time period** check box, you can set the current period to use any date category in the time dimension.

In general, you select a date category that appears at the lowest level in the time dimension, but this is not a mandatory requirement. Once set, each time category is generated and, before the creation of any cubes, the latest date category at the lowest level below the selected category is used to set the current period. This feature is useful when you want to automatically set the current period to the latest date in a year, rather than to the latest date in your data.

Steps

1. Open the property sheet for the time dimension and, on the **Time** tab, clear the **Automatically set the current time period** check box. Click **OK**, and open the **Categories** diagram for the time dimension.

2. Select the date category that you want to be the current period and, from the **Diagram** menu, click **Set Current Period**.

   The current period is set for the time dimension and automatically updated on the time dimension property sheet.

Track Changes in a Measure over a Specific Time Period

Businesses often need to track changes in a measure, from one time period to another.

For example, suppose your users want to gauge current sales revenues in light of previous performance. You set up the current period for your model to automatically use the most recent date values available after each data update. Then, you set up relative time categories that represent last month, last quarter, and last year, as well as the same month of the prior quarter, same quarter of the prior year, and same month of the prior year. Your OLAP report users can now analyze trends from month to month, quarter to quarter, and year to year.

Steps

1. Open the **Categories** diagram for the time dimension and position the pointer over the right side of the root category.

   The pointer changes to a crosshair.

2. Drag the pointer to the right of the root category.

   A new relative time category is created and its property sheet opens.

3. In the **Category code** box, type a name for the new category.

4. Click the **Relative Time** tab.

5. In the **Relative time** box, select the relative time period that represents the single period for which you want to create a relative time category, and click **OK**.

For a Year-Quarter-Month time dimension, the available single categories list includes the following:

- **Current Month**
- **Last Month**
- **Last Quarter**
- **Last Year**
- **Same Month, Prior Quarter**
- **Same Month, Prior Year**

Note: Select **Custom** if you want to override any of the default selections on the **Relative Time** tab. For example, you can specify the basic approach, a different target time period for your new category, a new context in which the period will be reported, and the corresponding offsets, to track values that apply to a period other than the current one.

For more information about specifying custom time periods, see "Track Changes in a Measure over Several Time Periods" (p. 109) and "Track Changes in a Measure in Future Time Periods" (p. 110).

Track Changes in a Measure for a Period-to-Date

Businesses often need to track changes in a measure for a specific period to-date.

For example, suppose your users want to gauge current sales revenues in light of previous performance. You set up the current period for your model to automatically use the most recent date values available after each data update. Then, you set up the following relative time categories: Current Month, Quarter To-Date, Year To-Date, Year To-Date, and Life To-Date.. Your OLAP report users can now analyze growth trends for a full range of to-date time periods.

Steps

1. Open the **Categories** diagram for the time dimension and position the pointer over the right side of the root category.

   The pointer changes to a crosshair.

2. Drag the pointer to the right of the root category.

   A new relative time category is created and its property sheet opens.

3. In the **Category code** box, type a name for the new category.

4. Click the **Relative Time** tab.

5. In the **Relative time** box, select the relative time period that represents the to-date period for which you want to create a relative time category and click **OK**.

   For a Year-Quarter-Month dimension, the available to-date categories list includes the following:

   - **Quarter To-Date**

- Year To-Date
- Life To-Date
- Quarter To-Date Grouped
- Year To-Date Grouped
- Quarter To-Date, Prior Quarter
- Year To-Date, Prior Year

Note: Select **Custom** if you want to override any of the default selections on the **Relative Time** tab. For example, you can specify the basic approach, a different target time period for your new category, a new context in which the period will be reported, and the corresponding offsets, to track values that apply to a period other than the current one.

For more information about specifying custom time periods, see "Track Changes in a Measure over Several Time Periods" (p. 109) and "Track Changes in a Measure in Future Time Periods" (p. 110).

Track Changes in a Measure over Several Time Periods

In some cases, you may want to set up relative time categories that span several specific time periods, at points in the past or the future. Several built-in relative time categories are available, to help or, if you have an unusual reporting period not covered by the automatically created relative time categories, you can create a custom category.

For example, suppose your users want to use relative time categories to track sales for prior months, quarters, and years. However, some users also want to track sales over six-month periods prior to the current date, in both the current year and the previous year. You set up an **N-period running total** category that spans the six months leading up to the current month, this year, and last year. Your OLAP report users can now analyze growth trends for the required time periods.

If you enter positive numbers for either the **Target offset** or the **Context offset**, your model must include source columns with time periods later than the current period, as positive values in these fields signal measures that are essentially forecasts, in future time periods.

Steps

1. Open the **Categories** diagram for the time dimension and position the pointer over the right side of the root category.

   The pointer changes to a crosshair.

2. Drag the pointer to the right of the root category.

   A new relative time category is created and its property sheet opens.

3. In the **Category code** box, type a meaningful name for the new category, such as Previous 6 Months.

4. Click the **Relative Time** tab.

5. In the **Relative time** box, select **Custom**.

6. In the **Basic approach** box, select **N-Period Running Total** or **N-Period Running Total (Grouped)**.

   Grouped lets you easily create a series of categories spanning different ranges of time.

7. In the **Number of periods** box, type the number of periods to include.

   For example, if you are creating a 6-month running total, type the number 6.

8. In the **Target period** box, select the type of period for which the N-period running total will be kept.

   For example, if you are creating a 6-month running total, select **Month**.

9. In the **Target offset** box, type a number that reflects an offset for the **Target period** relative to the current period.

   For example, if the current period is December 2006 and you want a 6-month running total up to (ending) November 2006, type -1.

10. In the **Context period** box, select a time period one or more levels higher than the **Target period**, within which you want to calculate the N-period running total.

    For example, if you are creating a running total of 6 months, create the total within the context of **Year**.

11. In the **Context offset** box, or in the **Context range** box if you selected **N-Period Running Total (Grouped)** in the **Basic approach** box, type the number by which the **Context period** is offset when the N-period running total is created.

    For example, if you are creating a 6-month running total relative to a **Target period** of last year, type -1.

    Your relative time dimension now contains the specified running-total time periods.

Track Changes in a Measure in Future Time Periods

If your source data contains forecast values (that is, source columns with time periods and date values that are later than the current period), you can add relative time categories such as Next Quarter or Next Year to your model to report future projections for the applicable measures.

For example, suppose the current period is December 31, 2006, but a data source containing sales forecasts for all four quarters of 2007 is included in the **Data Sources** list for the model. You create relative time categories with positive target or context offsets, to track the Next Year and Next Quarter projections.

Steps

1. Open the **Categories** diagram for the time dimension and position the pointer over the right side of the root category.

   The pointer changes to a crosshair.

2. Drag the pointer to the right of the root category.

   A new relative time category is created and its property sheet opens.

3. In the **Category code** box, type a meaningful name for the new category, such as Next Month.

4. Click the **Relative Time** tab.

5. In the **Relative time** box, select **Custom**.

6. In the **Basic approach** box, select the option that matches the kind of projection that you want.

   Depending on your selected approach, different controls appear on the **Relative Time** tab. For a current period of December, the settings for three typical future time periods are as follows:

Basic Approach: Category	Settings
Single Category: Next Month	**Target period**=Month; **Target offset**=1
Period To-Date Total: Year-to-date, Next Year	**To-Date Period**=Year; **Target period**=Month; **Target offset**=0; **Context period**=Year; **Context offset**=1
N-Period Running Total: First 6 Months, Next Year	**Number of periods**=6; **Target period**=Month; **Target offset**=-6; **Context period**=Year; **Context offset**=1

7. After you have set all the properties for the relative time category, click **OK**.

   Your relative time dimension now contains the specified future time periods.

Track Monthly Performance Measures in Different Time Periods

Business users often want to see monthly performance in one time period compared to another. You can add the extra time dimension to your model by using the calculated columns feature. This solution doesn't require a change to the date columns in your source data.

For example, you can use a calculated column to model monthly sales revenues, comparing the performance in 2006 to that in 2007. Your users can see a graphic representation of their results in their OLAP reporting component.

Steps

1. Add a calculated column to your model.

   For more information, see "Define a Calculated Column" (p. 85).

2. When prompted to select a data class for your calculated column in the **Column Data Class** dialog box, select **Numeric** and click **OK**.

3. In the **Column Calculation** dialog box, enter the following formula: `month(<Date>)` where `<Date>` is the column used to build your time dimension.

4. Click **OK** twice.

   The newly created calculated column appears as a new column in the data source.

5. Drag the new calculated column from the **Data Sources** list to the **Dimension Map** to create a new dimension.

6. Add the required levels, dimensions, and measures to your model and create the cube.

   For more information about creating cubes, see "Create a Single PowerCube" (p. 150).

Setting up Special Categories

A special category groups regular categories from any level in a dimension, without regard for their normal hierarchical organization. Special categories are unstructured and, unlike the categories in an alternate drill-down path, they must be maintained manually. However, your OLAP reporting component can be set up to show these categories in all drill-down lists, below the regular categories in the same dimension.

Note: In IBM Cognos 8, special categories are often considered alternate hierarchies within the same dimension.

By default, measure values are summarized. However, you can manually disable the rollup option.

If a cube is based on an apexed view, whether directly or indirectly by means of a cube group, the special categories become children of the apexed category.

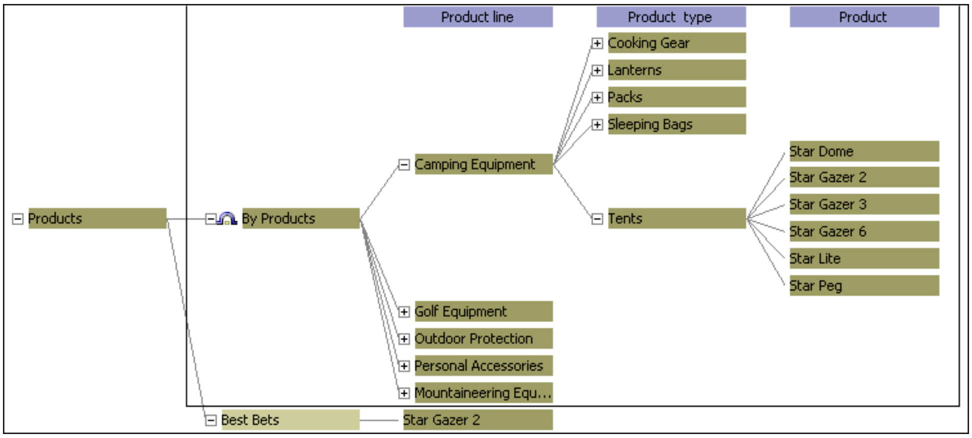
Create a New Special Category

To highlight important data in the model, you can create special categories.

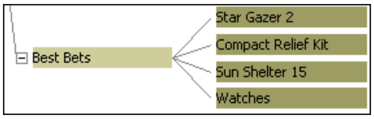
In the **Categories** diagram, the children of the special category appear both in the main hierarchy and in the special category branch. Any changes to a regular category within this group are reflected in both instances.

For example, suppose that one dimension in your model organizes all your products by Product Line and Product Type. You want to group your most promising products into a special category. These include Star Gazer-2 tents, the Compact Relief Kit first aid kit, SunShelter-15 sunblock, and all your Watches.

You create a Best Bets special category and drag the Star Gazer-2 category to it, as shown in this image.



You then drag the other promising product categories, in turn, from the Product and Product Type levels to your new category, yielding the following result.



Steps

1. Open the **Categories** diagram for the dimension for which you want to create the special category and position the pointer over the right side of the root category.

   The pointer changes to a crosshair.

2. Drag the pointer to the right of the root category and release the mouse button.

   A new special category is created and its property sheet opens.

3. In the **Category code** box, type the name of the special category and click **OK**.

4. Connect the special category to one or more regular categories by dragging the regular categories to the drop location of the special category.

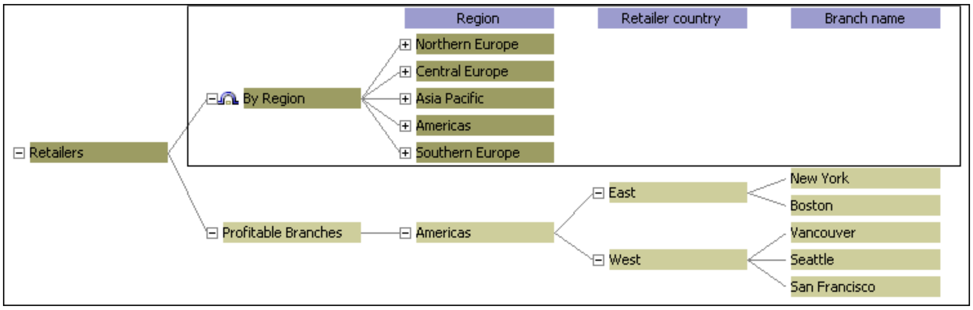
Create Special Category Levels

You can create drill-down levels by using special categories, thereby providing your users with an alternate drill-down path that uses a structure not supplied by your source data.

In the diagram, the children of the special category appear both in the main hierarchy and in the special category branch. Any changes to a regular category within this group are reflected in both instances.

For example, suppose you have a dimension called Region that includes branches from three countries in the Americas. You want to create a special category for your most profitable offices in this region and subdivide the category further by geography.

You begin by creating a special category named Profitable Branches and attach the Americas category to it. You create the lower-level categories East and West and then connect your most profitable branches appropriately; New York and Boston to the East, and Vancouver, Seattle, and San Francisco to the West.



Steps

1. Open the **Categories** diagram for the dimension for which you want to create the special category and position the pointer over the right side of the root category.

The pointer changes to a crosshair.

2. Drag the pointer to the right of the root category and release the mouse button.

   A new special category is created and its property sheet opens.

3. In the **Category code** box, type the name of the special category and click **OK**.

4. Add another special category to the right of the newly created special category.

5. In the **Category code** box, type the name of this special category and click **OK**.

6. Repeat steps 4 and 5 to add more categories at the appropriate level.

7. Connect the lowest level special category to the appropriate regular categories by dragging the regular categories to the drop location of the special category.

Disable Rollup on Special Categories

By default, measure values in special categories are summarized. However, you can disable rollup by changing the **Category rollup** setting on the appropriate **Special Category** property sheet.

If the **Category rollup** check box is cleared for a special category, it signifies that values from this category are not to be rolled up or summarized to the parent category.

It is possible for parents to add rollup values from some children and not from others. If no child categories have **Category rollup** selected, then the parent category becomes a placeholder and no values are associated with it when viewed in the OLAP reporting component.

Note: You cannot perform rollup on the highest level of a special category.

The **Category rollup** check box appears for special calculated categories. However, special calculated categories do not roll up to parent categories, even if this box is selected.

Steps

1. Open the **Categories** diagram for the dimension that contains the special categories.

2. Open the property sheet for the lowest-level special category.

3. Click the **General** tab, clear the **Category rollup** check box and click **OK**.

Resolving the Parentage of Orphan Categories

When you generate categories from your data source or automatically generate categories at cube creation time, Transformer reads and analyzes your source data and builds category trees for the dimensions and levels in your model.

Any source data that cannot be placed into a drill-down path are put in **Orphans Of** parent categories. By default, these categories, called source (or key) orphanages, are suppressed.

To eliminate source-category orphans, we recommend that you either remove the orphan categories from the source file or use drag and drop in the **Categories** diagram to associate them with their proper parents.

For example, suppose your model contains two data sources. One is from an order-tracking system, and supplies data for the Country, Region, Branch, and Customer Number categories. The other contains Customer Number and Address information. If the first source contains data for last year's orders and the second contains customer records from the two preceding years, any customer number not associated with an order in the period covered by the first source generates an **Orphans Of** category. This **Orphans Of** category should be resolved before you publish the PowerCube.

Resolve Invalid Date Errors

In most cases, Transformer can determine and apply the correct date format for input from a source column. An **Invalid Dates** category is automatically generated when the correct input format cannot be determined or when a problem arises with the source data.

You may have to change the format of the date information in your source file, change the **Date input format** property on the **Time** tab of the **Column** property sheet or, if the data is corrupt, rebuild your source file.

Steps

1. In the **Data Sources** list, open the property sheet for the column on which the time dimension is based.
2. Click the **Time** tab.
3. From the **Date input format** box, select the appropriate date format and click **OK**.
4. In the **Data Sources** list, select the data source for which you want to regenerate categories, and from the **Run** menu, click **Generate Categories From Selected Data Source**.

Adjust the Date Range to Encompass Early and Late Dates

Categories for **Early Dates** and **Late Dates** are automatically generated if dates from the source fall outside the range specified for any time dimension. By changing the date range, you can incorporate out-of-range dates into your model.

An **Early Dates** or **Late Dates** category may also be generated if the **Date input format** specified on the **Time** tab of a property sheet for a date column does not match the date format of the data in the source.

For example, suppose your data source contains regional sales data from 2006 to 2007. You set the **Earliest date** and **Latest date** on the **Time** tab of the time dimension property sheet to 20050101 and 20061231 respectively. A week later, the source is updated to include data for 2007. When you generate categories for your model, **Late Dates** categories appear. You follow the procedure outlined below to incorporate this new data into your model.

Steps

1. Open the property sheet for the time dimension and click the **Time** tab.
2. Type date values for **Earliest date** and **Latest date** as they currently exist in your data source and click **OK**.

3. In the **Data Sources** list, select the query that provides date values for the model.

4. In the **Data Sources** list, select the data source for which you want to regenerate categories, and from the **Run** menu, click **Generate Categories From Selected Data Source**.

Set up a Manual Level for Unknown Categories

You can create an orphanage category in a manual level, to keep new data separate until you can connect it to the appropriate parent category.

After you create an orphanage category, any subsequently generated categories that do not have a position defined in the model become children of that orphanage.

For example, suppose your organization is compiling customer satisfaction data for analysis by service, city, and region. The data is compiled over a two-week period, so the source is updated frequently. As you build the model, you want to store new data separately from the part of the model you are working on. You also want to be able to separately update the data in the completed areas of your model.

Steps

1. Open the **Categories** diagram for the dimension in which you want to create an orphanage category and click the highest (left-most) level, such as Region.

2. From the **Edit** menu, click **Insert Level**, and specify the **Level name** for your new orphanage level on the **Level** property sheet.

3. In the diagram, click on the right side of the drill category, so that the pointer changes to a crosshair, and drag the pointer to a position under the new level.

   When you release the mouse button, the **Category** property sheet appears.

4. Select **Category is an orphanage**, specify any other required category parameters and click **OK**.

Prevent New Categories from Being Added to a Dimension

If you do not want any new categories in your data source added to a particular dimension, you can prevent this from occurring automatically on category generation.

For example, suppose you have created a cube group based on one level in a dimension, and you do not want new categories added to that level; otherwise, each new category would automatically create a new cube for the cube group. You prohibit this action.

Steps

1. Open the property sheet for the applicable dimension.

2. Select **Prohibit automatic creation of new categories** and click **OK**.

Verify Your Model

Transformer provides a built-in validation tool to help you identify problems with your model design. We recommend that you run this tool after each major step in the model design process, to check for potential problems.

Also, if your model is based on more than one data source, you may find it useful to generate categories from a specific source, such as the one that supplies your key structural data or the values for your time dimension, rather than from all the data sources. You can generate categories from a selected data source with a simple **Run** command, without having to adjust the **Timing** controls for the cube on the **Data Source** property sheet.

Transformer automatically checks your model during data import, both before category generation and before cube creation. Certain warnings, including some related to auto-partitioning, do not appear with the checks done prior to cube creation.

Steps to Generate Categories from a Specified Data Source

1. In the **Data Sources** list, select the data source you want to use to generate categories.
2. From the **Run** menu, click **Generate Categories From Selected Data Source**.
3. Take note of any error messages. If a **Help** button is available, click it to review additional troubleshooting information.

Steps to Check the Validity of Your Entire Model

1. From the **Tools** menu, click **Check Model**.
2. Take note of any error messages. If a **Help** button is available, click it to locate additional troubleshooting information about the error, in the **Transformer Help**. For more information, see "Error Messages" (p. 279).

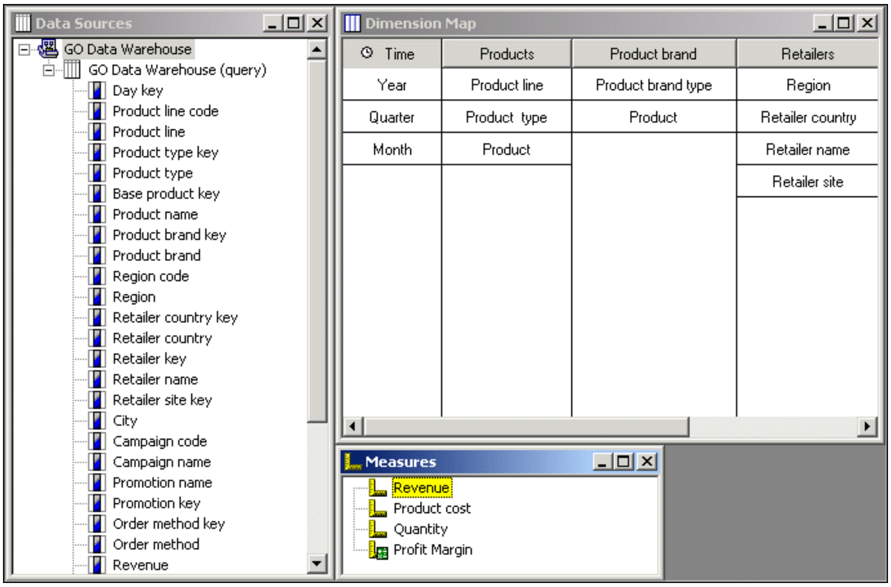
   Note: Transformer error messages are organized by ascending TR number.
3. If you are not able to resolve your model design problem on your own, contact Cognos Software Services for assistance.

Chapter 5: Modeling Your Key Performance Measures

In your Transformer model, you must include the key measures required to gauge the performance of your organization. Quantity of items sold and profit margin are examples of such performance indicators.

A model can contain any number of regular or calculated measures, together with logical groupings that use measure folders. A regular measure is numeric data in a transactional data source. A calculated measure is new numeric data that is derived from other measures, functions, and constants, such as Profit, which is calculated from the measures Revenue and Cost. A measure folder can group existing measures into a logical collection and, if needed, can be assigned a calculated value itself.

In this model, the **Measures** list contains both regular measures (Revenue, Product cost and Quantity) and a calculated measure (Profit Margin).



Time	Products	Product brand	Retailers
Year	Product line	Product brand type	Region
Quarter	Product type	Product	Retailer country
Month	Product		Retailer name
			Retailer site

To perform calculations on a measure, you create mathematical expressions based on an extensive list of available functions. For more information, see "Transformer Functions" (p. 126) and "IBM Cognos 8 Transformer Expression Editor" (p. 379).

Report users generally want to see their measure data summarized across one or more time periods, such as monthly or quarterly. You can apply different types of rollup to handle these summarizing operations, such as computing the mathematical average or taking the last result.

During the modeling process, you should also confirm how duplicate values are consolidated, and determine if any values need to be allocated to categories, levels, or dimensions that lack a direct association with these values. For more information about these important modeling processes, see "Controlling How Measures Roll Up" (p. 127), "Consolidation" (p. 134), and "Allocating Measures" (p. 137).

You should also confirm the scale and precision settings (p. 75) for measures when creating models based on report or package data sources. Precision settings determine the number of decimal places or significant digits in your measures. Transformer brings in scale and precision settings when they are available. However, when these settings are not available in the report or package query items, values defined for these settings in the data source may produce unexpected results. For example, values may produce excessive rounding or values may not scale correctly when building PowerCubes. In addition to checking the scale and assigning appropriate precision settings, ensure you verify that results for query and cube rollup operations in Transformer are displayed as you expect.

Add a Regular Measure

Each model contains one or more regular measures; that is, numeric data from a transactional data source, such as the number of units sold, the cost of goods purchased, or sales revenue.

The specified query items, including measures, are retrieved from the data sources specified in your model. After your cubes and data sources are published, authorized users can access the resulting cubes and OLAP reports for analysis.

For example, the following transactional data source has the regular measures Quantity, Revenue, and Cost. Each record represents a sales transaction.

Non-Measure dimensions			Measure dimensions		
DATE	STORE	PRODUCT	QUANTITY	REVENUE	COST
20060114	STORE1	TR139SQ	500	5000.00	4000.00
20060201	STORE1	TR139SQ	200	2000.00	1538.46
20060210	STORE1	TR139SQ	400	4000.00	3200.00

Step

- To add a regular measure to your model, drag the column from the **Data Sources** list to the **Measures** list.

Define a Measure Folder

When you use measure folders to create groupings of measures, your users can more easily navigate through various measure rollups and drill down to view the lower-level measures in their OLAP reports.

Measure folders can represent logical groupings of lower-level measures, or they can be calculated measures without any lower-level measures. That is, the calculation of a folder-level measure does not have to reflect the total value of the measures contained within that folder. The measures can be independent of each other from a calculation point of view.

For example, when your report users view the Profit Margin measure folder, they can drill down to see the lower-level measures: Revenue and Product Cost. The sum of these nested measures is reflected in the measure folder as a calculation. The detail of the calculated measure is hidden from users until they drill down.

If you create a measure folder that is not a calculated measure and does not contain any children, the **Check Model** tool shows a warning message, and that folder is not added to the cube.

Inclusion or exclusion of a measure on the **Measures** tab of the **PowerCube** property sheet does not cascade to the measures that exist below it. If you exclude a measure folder in the **Measures** list, the lower-level measures still appear. Each measure must be excluded individually.

Note: Measure folders in IBM Cognos 8 Report Studio and Analysis Studio do not behave the same way as they do in IBM Cognos Series 7 PowerPlay. You cannot drag and drop measures under a measure folder by dragging the measure folder itself. Instead, select all measures under the measure folder and drag them into the report or analysis.

Steps to Add a Measure Folder

1. Right-click the **Measures** pane and click **Insert Measure Folder**.
2. In the **Measure name** box, type a name for the measure folder, and click **OK**.

   The measure name must be unique in the model. By default, the **Short name** and **Measure label** options derive their values from the measure name because they are set to **<<From Measure Name>>**.

   Tip: Use the original name of the associated source column to make a regular measure name unique.
3. Drag existing measures into your new measure folder.

Steps to Change a Calculated Measure to a Measure Folder

1. Right-click the measure and click **Properties**.
2. On the **General** tab, select the **Measure Folder** check box, and click **OK**.

Define a Measure that Counts Categories

You can define a measure that counts the categories present in a unique level. To be counted, the categories must have non-missing, non-zero values and be associated with a supported **Activity measure** type.

For example, you can create a count to show how many customers of each type bought a specific product each month, quarter, or year. This count is a consolidation of all the categories in the unique level Customer Number. For example, if the same customer buys a product in two months, the quarterly rollup counts that customer only once.

Constraints

You should accept the default activity measure **All Measures** only if you are sure that no categories are missing measures, and no measure values sum to zero when they are rolled up to the next higher time period.

You must use the **Rollup** tab of the **Measure** property sheet to specify the **Activity measure** for the category count. This **Activity measure** cannot be any of the following:

- a before or after-rollup calculated measure

  For more information, see "Define When Measures are Calculated" (p. 124).

- an externally rolled up measure

- another category count measure

- an allocated measure, if the allocation occurs in the same dimension as the category count and above the counted categories level

An error message appears if you try to specify any one of these restricted types of measure as the basis for your count.

You can base category counts on allocated measures if the allocation uses a constant from a dimension other than the category count dimension. You can also create category count measures in time-based partitioned cubes, subject to certain restrictions. To see information about time-based partitioning, consult the index.

Steps

1. Click the **Measures** list to make it active and, from the **Edit** menu, click **Insert Measure**.

2. In the **Measure name** box, enter a name for the measure, and click **OK**.

   The measure name must be unique in the model. By default, the **Short name** and **Measure label** options derive their values from the measure name because they are set to **<<From Measure Name>>**.

   Tip: Use the original name of the associated source column to make a regular measure name unique.

3. Select the **Type** tab and click **Category Count**.

4. Select an entry from the **Dimension** and **Level** drop-down lists.

   The level must be unique and situated in the primary drill-down path of the dimension you have selected.

   Tip: Confirm that the **Unique** option is selected on the **Level** property sheet, and that it is true for your data, by checking the scope. For more information, see "Show the Scope for a Measure" (p. 138).

5. Select the **Rollup** tab and then select a supported **Activity measure** for the count.

6. Define any other attributes of the count, such as the format and click **OK**.

The category count appears in the **Measures** list of your model. The resulting cube and OLAP reports will show the number of non-missing, non-zero categories for the specified level and **Activity measure**.

Define a Calculated Measure

You can create a calculated measure to derive new numeric data from regular measures, other calculated measures, functions, and constants.

You can specify the format for the result, and also explain the calculation in the **Description** box on the **Measure** property sheet, so that your users can see how the data was derived.

For example, you create a calculated measure for Profit Margin by subtracting Cost from Revenue and dividing the result by Revenue. You select **0%** on the **Format** tab, so that the results of your calculated measure are expressed as percentages. For an item costing 2000 with a revenue of 4000, your users see a Profit Margin of 50%.

After you create a calculated measure, you can control whether the measure is calculated before or after rollup.

All calculations defined in a cube are performed based on built-in rules of precedence. Where there are intersecting calculations, the after-rollup calculations defined for all measures are performed first. These are followed by calculations for calculated categories, and then by dynamic and regular rollups. This is important information for your users, who need to understand the results they are seeing in the intersecting cells of their crosstab reports.

Steps

1. Click the **Measures** list to make it active and, from the **Edit** menu, click **Insert Measure**.
2. In the **Measure name** box, enter a name for the measure.

   The measure name must be unique in the model.
3. Select the **Type** tab, click **Calculated** and then click **Calculation**.
4. In the left pane of the expression editor, expand the **Functions** and **Measures** folders as needed, select each parameter you want to use and click the right-arrow button to insert it into the **Expression definition** box in the right pane of the editor.

   For more information, see "Transformer Functions" (p. 126) and "IBM Cognos 8 Transformer Expression Editor" (p. 379).
5. When the expression is complete, click **OK**.
6. Type or select the other features you want for the measure, such as the description and format and click **OK**.

   Your new calculated measure appears in the **Measures** list.

Define When Measures are Calculated

When you use a calculated measure, you can have the measure calculated before or after rollup. Values for measures calculated before rollup are based on source values instead of summed values, and are rolled up separately.

The following table shows the sequence of operations that take place when calculating measures.

Timing	Operation Sequence
After Rollup	1 Regular rollup takes place on all measures. 2 Time state rollup takes place. 3 Allocated measures are calculated. 4 Calculations are performed on calculated measures.
Before Rollup	1 Calculations are performed as the cube is created. 2 Regular rollup takes place on all measures. 3 Time state rollup takes place. 4 Allocated measures are calculated.

Although values for measures calculated after rollup require less storage space, time-state rollup cannot be performed on **After Rollup** calculated measures.

Auto-partitioning is not used when a model contains **Before Rollup** measures, which can be detrimental to run-time performance. To allow auto-partitioning to proceed, we recommend that you use calculated columns instead of **Before Rollup** measures wherever possible.

Similarly, although there is no consolidation of **Before Rollup** measures, you can use calculated columns to perform consolidation instead. The measure data is then consolidated as each row is read into the work file.

Steps

1. Open the property sheet for the calculated measure and click the **Rollup** tab.
2. In the **Regular timing** box, click **Before Rollup** or **After Rollup** and click **OK**.

Specify How Missing Values Appear

Transformer uses default measure settings for missing values to determine how null results are handled. In Transformer version 8.x, the default setting for missing values is **NA**; when the default setting is used, **na** is inserted into your OLAP reporting component for the missing values. In releases of Transformer prior to version 8.3, the default setting for missing values was **Zero**; when that default setting was used, zeros were inserted into the OLAP reporting component for the missing values.

The default setting can also affect the values that are stored and shown for calculated measures. When a calculated measure is based on a measure for which the **Missing value** setting is **Zero**, and there is no data for this second measure, the calculation will be valid because the missing information is treated as a zero. However, when the missing value setting is the Transformer version 8.x default, **NA**, the calculation result will be invalid because the missing information is treated as a null value and not a zero.

When models created with previous versions of Transformer are upgraded to Transformer version 8.x, measures that have the missing value settings set to the Transformer version 7.x default (**Zero**) will not be changed to the Transformer version 8.x default (**NA**). Upgraded models will handle missing values in the same way as they did in previous versions of Transformer.

To ensure that missing values are shown appropriately, review the missing value settings for all measures before building new cubes.

Steps

1. Open the **Measure** property sheet and click the **General** tab.
2. In the **Missing value** box
   - Click **Zero** to have missing values appear as zeros.
   - Click **NA** to have missing values appear as NA.
3. Click **OK**.

Reverse the Sign of Measure Values in Financial Models

You can reverse the sign of measure values for non-root categories if the sign would not otherwise make sense to the intended audience.

For example, your accounting system stores revenue and liability balances as negative values, and expense and asset balances as positive values. You want to distribute reports based on this data to sales managers, who logically expect revenue values to be positive. To accommodate the information needs of this audience, you reverse the sign for revenue values from negative to positive.

Reversing the sign does not affect the value properties, only how they appear in your OLAP reporting component. However, your report users should be advised that sign reversal can alter the results of run-time calculations. We recommend that you state which categories contain reversed signs in the **Description** box of the **Measure** property sheet, thereby ensuring that this important information is shared with the appropriate users.

To reverse the sign of measure values for specific categories, you must first identify the measure to be reversed and then select the categories for which the reversal applies.

Steps

1. Open the **Measure** property sheet for the measure whose sign will be reversed and click the **General** tab.
2. Select the **Reverse the sign** check box and click **OK**.

This option identifies the measure whose values will be reversed once the target categories have been selected and the cube is generated.

3. Open the **Categories** diagram for the dimension containing the categories you want to change.
4. Open the **Category** property sheet and click the **General** tab.
5. Select the **Reverse the sign** check box and click **OK**.
6. Repeat steps 4 and 5 for each category whose sign you want to change.

Transformer Functions

The Transformer expression editor supports a common set of mathematical calculations and a few additional Transformer-specific functions, as explained below.

Note: The Transformer expression editor is not the same expression editor used in Framework Manager or the IBM Cognos 8 Web studios.

When you build an expression, you select elements, one by one, from the **Available components** list in the left pane of the expression editor and build your expression in the right pane. The available elements vary according to the type of calculation and the position in the expression. Explanations for each element are provided in a **Tips** window in the expression editor and, as you enter each element, the syntax is evaluated against standard mathematical rules.

Tip: If the Tips box is not visible, click the **Tips** button ⓘ.

You can also copy or type your calculation directly into the **Expression definition** pane.

The supported functions for calculated measures are:

- **absolute**

  Returns the absolute value.

- **average**

  Returns the average value.

- **min**

  Returns the minimum value of selected data items.

- **max**

  Returns the maximum value of selected data items.

- **percent**

  Returns the percent of the total value for the selected data items.

The supported arithmetic operators are **+** (addition), **-** (subtraction), ***** (multiplication), **/** (division), and **^** (exponentiation).

The Transformer expression editor also supports if-then-else constructs and brackets. The boolean portion of the if-then-else supports AND, OR, and isnull (a function that tests if a measure value is empty).

For more information about the expression editor, see "IBM Cognos 8 Transformer Expression Editor" (p. 379).

Controlling How Measures Roll Up

The rollup function specifies how measure values are summarized, or rolled up, from child categories to their parent categories. Summarization occurs when your OLAP reporting component displays measure values in categories above the lowest level of detail.

When the type of rollup used is specified in the **Description** box on the **Measure** property sheet, other users can reference this information. The three kinds of rollup are as follows:

- **Regular rollup** combines records with identical values in non-measure columns. It is applied to the time dimension by default unless the **Time state rollup** option is specified for the measure.

- **Time state rollup** combines records with identical values in the time dimension according to the **Time state rollup** function selected.

  For example, level of inventory is recorded for a specific product, at the same warehouse, in the same year, on different dates. If you select **Average** and the **Degree of detail** set for the column is **Day**, the identical records are summed and then divided by the number of day categories in the month. However, if you select **Last period**, the inventory value for the last day of the month is shown.

- **Duplicates rollup** is automatically used whenever the PowerCube **Consolidate** feature is turned on.

If you select both **Regular rollup** and **Time state rollup**, regular rollup is performed first, followed by the time state rollup.

If the measure is used as a weighted value, as occurs during measure allocation, you can use only the **Sum** rollup type.

When a model has more than one time dimension, you cannot use time state rollup.

Set a Regular Rollup Function for Measures

By default, the values for a measure are automatically totalled using the **Sum** function. Depending on the context, you may change this default rollup to **Minimum**, **Maximum**, **Average**, **Count**, **Count All**, **Any**, or **External**.

For example, suppose your sales analysis model contains a measure named QTY. You want your OLAP reports to show the average monthly quantity sold for each store. You use **Regular rollup** (the **Average** function) to obtain the correct result. Your source data is as follows.

Date	Store	Product	QTY
20070101	STORE1	TR139SQ	500
20070131	STORE1	TR139SQ	200

Date	Store	Product	QTY
20070101	STORE2	TR139SQ	400
20070131	STORE2	TR139SQ	600

After applying the **Average** rollup function, the following results appear in your report.

	STORE1	STORE2	STORE
2007	350	500	425
Date	350	500	425

Calculated measures can be computed after rollup or before rollup. When the measures are calculated before rollup, the values are those calculated in the data source.

For the time dimension, when you select **Average** as your regular rollup type, the records are first summed and the result is then divided by the number of records in the rollup time period.

The formula for a first-quarter average involving 78 records can be represented as follows:

$$\frac{\sum \text{Jan} + \sum \text{Feb} + \sum \text{Mar}}{20 + 3 + 55}$$

Steps

1. Open the **Measure** property sheet and click the **Rollup** tab.
2. In the **Regular rollup** box, select a rollup function.

   If you select the **Average** function, you can weight the average by selecting a measure from the **Regular weight** box. However, the weighting measure must have a rollup type of **Default** (**Sum**) or **Sum**.
3. Click **OK**.

Create Cubes with External Rollups

In Transformer, you use the rollup function to summarize measure values in the cube. If you have specific data that you do not want changed within the cube, you can use externally rolled up measures to maintain control over which values are stored and displayed.

To create a cube with externally rolled up measures, you first define which measures are to be externally rolled up, you then create the required date categories and, finally, you specify the **Degree of detail** for allocations, if applicable.

Externally rolled up measures can be used with alternate drill-down paths, special categories, and partitioned cubes, including those partitioned on the time dimension.

Each data record in an externally rolled up transactional data source uses a category code to reference the category in the model. To avoid ambiguity and unpredictable results, ensure that each code uniquely identifies a specific category in the source file. Otherwise, Transformer makes the code unique by adding a tilde (~) character with a numeric suffix, creating a blended expression that cannot be interpreted by other IBM Cognos 8 components.

For example, suppose your cube contains more than one instance of the category code Item, and so renames the second instance Item~1, the third instance Item~2, and so on. These category codes are numbered as they are encountered in the model. This means that the codes do not stay the same, in a predictable sequence, as your organization (and model) evolves. The tilde characters are interpreted as non-numeric and, because these codes do not appear in the data source, they are ignored during the consolidation pass. As a result, duplicate records are overridden and not summed, despite what may be reported in the log file.

To avoid such problems, use one of the following strategies to make these codes unique:

- Ensure that all source values are unique in a dimension.
- Create a calculated column in Transformer, to make your categories unique.
- Edit the .mdl model file to make the category codes unique.

  For more information about using the Model Definition Language (MDL), see the Transformer *Developer Guide*.

For each special category that has more than one child category, you must provide the externally rolled up value. Otherwise, the special category value will be missing and, by default, be reported as zero. If the special category has only one child, you can either supply a value for the special category or accept the value taken from the child.

Steps

1. After you have imported the structural and transactional data sources into your model, open the **Data Source** property sheet for the transactional data source and click the **General** tab.
2. Click the **Contains externally rolled up measure values** check box and click **OK**.
3. Add the measures to the **Measures** list.
4. Open the **Dimension** property sheet for each dimension that is to have externally rolled up measures applied to it and click the **General** tab.
5. In the **External rollup column** box, click **Add** to specify which transactional column contains the category codes the measure values will map to.
6. In the **New Association** dialog box, select **Source** from the **Association role** drop-down list, and click **More** to select the external rollup column. Click **OK** three times.
7. For each externally rolled up measure, open the **Measure** property sheet and click the **Rollup** tab.
8. In the **Regular rollup** box, click **External**.

   Note: For date dimensions or dimensions without unique source values, specify the column from which category codes can be assigned in order to make these codes unique in the dimension, as follows:

- Open the **Level** property sheet for each level in each dimension with externally rolled up measures.
- In the **Associations** box, select the source column and click the ellipsis button (**...**) to open the **New Association** dialog box.
- In the **New Association** dialog box, select the **Association** role, and then click the **More** button to select the structural source column to map to the level.
- If you are using allocations, on the **Column** property sheet, set the **Degree of detail** to specify the lowest level to which the associated measure applies.

9. Create the cube by clicking **Create PowerCubes** from the **Run** menu.

Set a Time State Rollup Function for Measures

The time state for a measure is its value at a specific point in time, such as the inventory quantity of an item at the end of the month. You set the **Degree of detail** property to the lowest level of detail for the transaction; that is, the values at the bottom of the hierarchical tree. For example, if your source file contains daily transactional records, the degree of detail is **Day**.

You can use any of the following **Time state rollup** functions to summarize the values from your non-measure columns: **Minimum**, **Maximum**, **First period**, **Last period**, **Current period**, and **Average**.

Constraints

There are several constraints that apply when defining time-state rollups:

- **Regular rollup** is not applied to a time dimension if **Time state rollup** is selected. Also, you cannot apply summarized views to a time dimension when the model contains a measure with **Time state rollup** applied.
- **Time state rollup** cannot be applied to a measure if the model has more than one time dimension.
- If you use **Time state rollup** but more than one record is associated with each data point, values will be summarized using **Regular rollup**. For example, if there are two inventory stock counts daily, the two records will be summed even though you specified a **Time state rollup** function of **Last period**.
- You can use calculated columns to consolidate multiple measurements. If you do, you should also set the **Regular Timing** option to **Before Rollup**. Otherwise, the measures are calculated after rollup (the default), which yields inaccurate results.
- If a record is missing, you may get zeros when you apply a rollup. For example, if you select a time state rollup of **Last period**, but your source data does not contain records for every day of every month, then whenever values are missing for the last day of the month, the OLAP report will show a zero.
- A time-state rollup of **Average** may also result in zeros, but you can specify that any missing values display as **NA** (or **na**) in the OLAP reporting components.

- If you select a time-state rollup of **Average**, the **Days in week** selection affects the results. If the **Degree of detail** is **Day**, and your **Days in week** selection excludes weekends, the records for each month are summed and then divided by the number of categories (non-weekend days) in that month.

$$\frac{\sum \text{Jan} + \sum \text{Feb} + \sum \text{Mar}}{23 + 21 + 21}$$

Steps

1. Open the **Measure** property sheet and click the **Rollup** tab.

2. In the **Time state rollup** box, select a rollup function.

   If you select the **Average** function, you can weight the average by selecting a measure from the **Time state weight** box. However, the measure associated with the weighted values must have a **Rollup** function of **Default** (**Sum**) or **Sum**.

3. Click **OK**.

Example - Setting a Last Period Time State Rollup for Inventory Counts

Your source data contains warehouse inventory counts for the middle and end of each month. You want an OLAP report that shows inventory at the end of each quarter.

Your source data is as follows:

Date	Product	QTY
20060915	TR139SQ	500
20060930	TR139SQ	200
20061015	TR139SQ	300
20061031	TR139SQ	300
20061115	TR139SQ	400
20061130	TR139SQ	600
20061215	TR139SQ	250
20061231	TR139SQ	350

You select **Last period** from the **Time state rollup** drop-down list of functions for the quarter date level.

Your OLAP report shows the following quarterly data, and the year-level rollup shows the last inventory count recorded.

Date	Product	QTY
200609 (end of 3rd quarter)	TR139SQ	200
200612 (end of 4th quarter)	TR139SQ	350

Ignore Null and Missing Values in Specified Time State Rollups

You can specify that null and missing values be ignored when applying average or weighted average time-state rollups.

You can achieve the same result using the Model Definition Language (MDL), by setting the `IgnoreMissingValue` keyword to `TRUE` when you create or update the definition for a supported measure type.

However, Transformer only supports this feature if you specify that missing values be treated as **NA** on the **Measure** property sheet. Also, you must retain the default setting for **First period, Last period**, and **Current period**. That is, null and missing values cannot be excluded from the rollup calculations for these measure types.

Steps

1. Open the **Measure** property sheet and click the **General** tab.
2. In the **Missing value** box, select **NA.**
3. Click the **Rollup** tab.
4. Select the **Ignore missing values in average and weighted average time-state rollups** check box and click **OK.**

   Note: If the rollup measure is of type **First period, Last period**, or **Current period**, the **Ignore missing values in average and weighted average time-state rollups** check box is disabled. Missing (null) data values are always excluded from **Minimum** and **Maximum** calculations for rollups, whether they are set to display as **0** or **na** in the OLAP reporting components.

Set Regular and Time State Rollup Together

You can use **Regular rollup** and **Time state rollup** together to summarize measure values in your OLAP reports. When both are selected, regular rollup is performed first, followed by time-state rollup.

Constraints

There are several constraints that apply when defining these two types of rollup together:

- Transformer cannot perform **Time state rollup** for a measure when there is more than one time dimension in the model.
- You cannot apply summarized views to a time dimension if the model contains a measure with **Time state rollup** applied.

- For the time dimension, if you select **Average** for both **Regular** and **Time state rollup**, Transformer performs **Regular rollup** first. It then sums the remaining consolidated records and divides by the number of leaf categories in the time period. Lastly, it sums the remaining records and divides by the number of months in the quarter or days in the month.

Steps

1. Open the **Measure** property sheet and click the **Rollup** tab.

2. In the **Regular rollup** box, select a rollup function.

   If you select the **Average** function, you can weight the average by selecting a measure from the **Regular weight** box. However, the weighting measure must have a rollup function of **Default** (**Sum**) or **Sum**.

3. In the **Time state rollup** box, click a rollup function.

   If you select the **Average** function, you can weight the average by selecting a measure from the **Time state weight** box. However, the weighting measure must have a rollup function of **Default** (**Sum**) or **Sum**.

4. Click **OK**.

Example - Jointly Setting Regular and Time-State Rollup

Your source data contains the inventory levels for the middle and end of each month in all warehouses. You want your OLAP report to show the total inventory at the end of each quarter. To obtain the correct result, you define a view that removes the warehouse category, and then use **Regular rollup** (**Sum**) and **Time state rollup** (**Last period**).

Your source data is as follows:

Date	Warehouse-Product	QTY
20061115	WH1TR139SQ	500
20061115	WH2TR139SQ	400
20061130	WH1TR139SQ	200
20061130	WH2TR139SQ	600

You select **Regular rollup** for each period, and then apply **Time state rollup**.

The first rollup, **Sum**, yields QTY values of 900 for 20061115 and 800 for 20061130. The second rollup, **Last period**, yields a QTY of 800 for 200612.

Consolidation

Consolidation uses rollups to combine records with identical non-measure values into a single record, thereby reducing cube size and improving run-time performance in your OLAP reporting component.

Records may have identical non-measure values in the following circumstances:

1. The source contains transactions with identical non-measure values.

   For example, two sales of the same product are made to the same customer on the same day.

2. The degree of detail permits it.

   For example, the **Degree of detail** for a column associated with the time dimension is set to **Month**, so **Day** values in the source transactions are ignored during consolidation.

3. A dimension is omitted from the cube.

   For example, two sales of the same product are made at different stores on the same day. If stores are omitted from the cube, the sales records have identical non-measure values.

4. Categories in the cube are summarized or suppressed.

   For example, two sales of the same product are made to the same customer on the same day, but the colors differ. If colors are omitted from the cube by using either of these dimension view options, the sales records have identical non-measure values.

In cases 1, 2 and 3, consolidation uses **Duplicates rollup** to combine records with identical values in their non-measure columns. In case 4, unless **Time state rollup** is selected on the **Measure** property sheet, consolidation uses **Regular rollup** to combine records with values made identical through the use of dimension views.

Notes

When consolidating data, the following additional considerations apply:

- A cube will not be consolidated if **Time state rollup** is defined for it, or if other roll-up actions will conflict with consolidation.

- Consolidation occurs automatically if cubes use auto-partitioning.

- You can specify which type of consolidation to use: either **Yes** (**with sort**) or **Yes** (**presort**).

- The order in which rollups are done affects cube output: **Duplicates rollup** precedes **Regular rollup**.

Tip: To combine records without affecting the cube, you can use **Regular rollup** and **Time state rollup** without consolidation. These combinations only affect how measure values are aggregated at run time. For example, you can use a **Last period** time state rollup for inventory data, so that your reports show only the last measure values for each time period.

Example - Consolidating Data Using Duplicates Rollup (Sum)

You want to specify that duplicate records be summed as they are rolled up through the levels in your time dimension.

Suppose your source data contains daily transaction records such as the following, which are used to create the Date and Region dimensions.

Date	Region-Store	Quantity	Revenue
20070105	East-Store1	50	5000
20070110	East-Store2	20	2000
20070131	East-Store1	40	4000
20070201	East-Store1	60	3000
20070228	East-Store1	30	4000
20070305	East-Store1	60	3500
20070315	East-Store2	40	4000

Because the **Degree of detail** is set to **Month**, day values are ignored. Records for the same month and store are consolidated (summed).

The results after **Duplicates rollup** (**Sum**) are as follows.

Date	Region - Store	Quantity	Revenue
200701	East - Store1	90	9000
200701	East - Store2	20	2000
200702	East - Store1	90	7000
200703	East - Store1	60	3500
200703	East - Store2	40	4000

After **Regular rollup** (**Average**), the monthly records for each Store in the East region are consolidated again.

Date	Region	Quantity	Revenue
200701	East	110/2=55	11000/2=5500

Date	Region	Quantity	Revenue
200702	East	90	7000
200703	East	100/2=50	7500/2=3750

Your cube has a dimension view where store values are aggregated to the East level.

Set a Duplicates Rollup Function for Measures

The **Duplicates rollup** function specifies how records containing identical non-measure values (category names) are aggregated. The records may or may not have different measure values.

The default **Duplicates rollup** for each measure is **None** (**Regular rollup**). However, you can change the setting to **Sum**, **Minimum**, **Maximum**, **Average**, **First**, or **Last**.

You can explicitly request consolidation by changing the **Consolidate** setting on the **General** tab of the **PowerCube** property sheet.

If values for calculated measures are generated before rollup, Transformer cannot perform consolidation. However, you can overcome this problem by using calculated columns. Because such values are always calculated before rollup, consolidation is again possible.

Steps

1. Open the **Measure** property sheet and click the **Rollup** tab.
2. In the **Duplicates rollup** box, select a rollup function, and click **OK**.

   If you select **Average**, you can weight the average by selecting a measure from the **Duplicate weight** box.
3. If you have not yet built your cube, from the **Run** menu, click **Create PowerCubes**.
4. Open the property sheet for your cube and click the **General** tab.
5. In the **Consolidate** box, click **Yes** (**with sort**) or **Yes** (**presort**).

Example - Consolidating Data Using a Duplicates Rollup of Average

Although duplicate categories have their measure values summed by default, in some cases, you may want to specify a different rollup function, such as **Average**.

For example, suppose the following before-rollup records are duplicates.

Date	Store - Product	Quantity	Price	Revenue
20070101	Store1 - TR139SQ	50	100	5000
20070101	Store1 - TR139SQ	20	100	2000
20070101	Store1 - TR139SQ	40	100	4000

Date	Store - Product	Quantity	Price	Revenue
20070101	Store1 - TR139SQ	70	80	5600

The results after **Duplicates rollup** (**Average**) are as follows:

Date	Store-Product	Quantity	Price	Revenue
20070101	Store1 - TR139SQ	180/4=45	380/4=95	16600/4 = 4150

Allocating Measures

The **Allocation** feature distributes data, specified at a summary level of a dimension, to lower levels. For example, actual sales revenue may be tracked daily, while sales revenue is forecast quarterly. Allocation is a useful way of distributing quarterly forecasts to the month and day levels.

You can select from the following allocation types:

- **From Level**

  With this type, allocation is typically based on values directly derived from a data source.

- **Do Not Allocate**

- **Constant**

  This type allocates a single, constant measure value to all descendant categories.

- **By Measure**

  This type allocates values to descendant categories in proportion to a specific measure.

You can allocate measures to dimensions or levels that lack them in the following ways:

- over entire dimensions, when the measure appears in a data source that does not reference the dimension

- over levels in a dimension, when the measure is already specified at a level in that dimension

- over categories in levels, when the measure is specified for the particular level

To allocate measures, you must

- Have a model with two or more data sources.

- Define the measures for that model.

- Create the **Dimension Map**.

When you have completed the allocation process, you can check the relationships between dimensions and a measure by using the **Show Scope** command.

You cannot allocate a calculated measure. However, you can allocate a regular measure in proportion to a calculated measure.

To prevent the automatic rollup of constantly allocated measures, you must set a special variable so that the totals in your summary reports are the same as the constants specified for each individual member of a subset.

To do this on Windows, locate the cogtr.xml file in the *installation_location*/configuration directory, open it in any text editor, and add the following entry to the [PowerPlayDataServer] section: `DISABLE_CONSTANT_ALLOCATION_ROLLUP=1`.

If you are adjusting your model on UNIX or Linux, create an environment variable called `PPDS_DISABLE_CONSTANT_ALLOCATION_ROLLUP` and set its value to `1`. In Bourne shell, add the line export `PPDS_DISABLE_CONSTANT_ALLOCATION_ROLLUP`.

Note: This change will affect other features and functions in your model or resulting reports, including calculated expressions, alternate or special drill-down paths, and custom subsets.

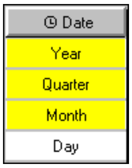
Show the Scope for a Measure

When you run the **Show Scope** command, you see a color-coded **Dimension Map** that highlights relationships between the measure and levels in a dimension. This scope map helps you identify where a measure is meaningful in models based on multiple data sources.

The scope of a measure depends on its relationship to the levels in the **Dimension Map**.

Level Derived Directly

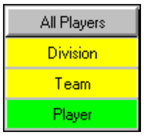
In this case, the dark yellow color indicates that the level takes its category values directly from the measure.



In this example, the levels Year, Quarter, and Month are directly associated with the measure, while the lowest level, Day, is associated with another measure.

Level with Allocated Measures

In this case, the green color indicates that the level values are allocated either by a constant or proportionally to another measure.



In this example, the dimension contains a level named Player, whose measure values are allocated proportionally to an ancestor level.

Calculated Measure Scope

If a calculated measure is based on two regular measures, the measure scope dimension map shows the lowest common level. If the lowest common level is allocated, the level appears with the same shading (green) for both the allocated measure and the calculated measure.

Steps

1. In the **Measures** list, select the measure whose scope you want to see.
2. From the **Edit** menu, click **Show Scope**.

   Tip: You can change the default colors of the scope map on the **Dimension Map** tab in the **Preferences** property sheet on the **File** menu. Click the color swatch you want to change and select a new color from the **Color** palette. Click **OK** twice.

Set Dimension Allocation

Transformer automatically allocates a measure as a constant throughout an entire dimension when the source of the measure does not reference that dimension.

However, you can change this default allocation to **Do Not Allocate** or **By Measure** (proportionally allocate, based on values in another measure).

Steps

1. Open the property sheet for a dimension that is not referenced by the measure you want to allocate and click the **Allocation** tab.
2. Right-click the **Allocation type** for the measure and select an option.

   Tip: You may also allocate measures to an entire dimension from the **Categories** diagram by double-clicking the **Root** category.

Set Level Allocation

When a measure applies to only some levels in a regular or time dimension, you can allocate the measure from a level where it applies to all lower levels that have no relationship established.

When you allocate a measure from a level, all categories in the level inherit the allocation type. However, you can change the allocation type of individual categories.

Steps

1. In the **Dimension Map**, double-click the parent level of the levels to which you want to allocate measure values, to open its property sheet.
2. Click the **Allocation** tab.

   A list of measures that are candidates for allocation appears.
3. In the **Measure list** box, right-click the **Allocation type** for the measure, and select an option.

   The allocation types include **Do Not Allocate**, **Constant**, and **By Measure**.

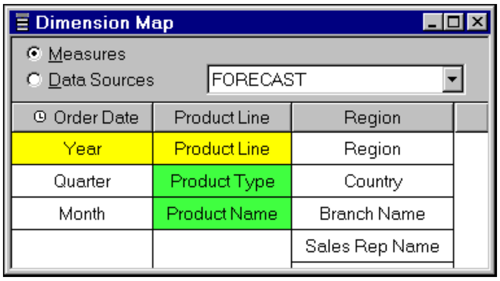
Tip: You can check the relationships between dimensions and a measure by using the **Show Scope** command on the **Edit** menu.

Example - Allocating a Measure to Levels in a Regular Dimension

You want to allocate sales forecasts, which are created at the Product Line level, to Product Type and Product Name levels, in proportion to actual revenues. The revenues are recorded daily, by product and region.

Using the **Level** property sheet for Product Line (the parent level), you set the **Allocation type** for the**Revenue** measure to **By Measure**.

You check the scope map to confirm your result.



● Measures
○ Data Sources
FORECAST

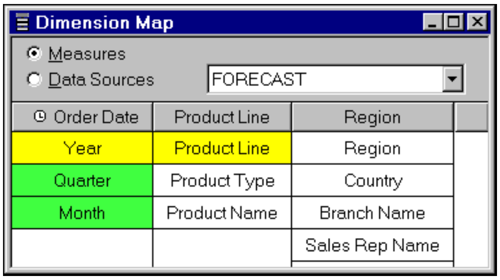
Order Date	Product Line	Region
Year	Product Line	Region
Quarter	Product Type	Country
Month	Product Name	Branch Name
		Sales Rep Name

Example - Allocating a Measure to Levels in the Time Dimension

You decide to allocate your annual sales forecasts on a monthly basis, in proportion to actual revenues.

Using the **Level** property sheet for Year, the parent level of Quarter and Month, set the **Allocation type** for the Revenue measure to **By Measure**.

You check the scope map to confirm your result.



● Measures
○ Data Sources
FORECAST

Order Date	Product Line	Region
Year	Product Line	Region
Quarter	Product Type	Country
Month	Product Name	Branch Name
		Sales Rep Name

Set Category Allocation

When you change the allocation type for a level, the new allocation type is applied from the categories in that level to all descendant categories. You can also set the allocation type for an individual category.

For example, suppose that some operating costs associated with your product lines are based on sales channel factors, such as direct sales and catalog orders. Others are based on revenue. You want to allocate costs for each product line proportionally to each cost factor. Using the **Category** property sheet, you set the allocation type individually for each product line category.

Steps

1. Open the **Categories** diagram, and then open the property sheet for the category for which you want to set the allocation type.

2. Click the **Allocation** tab.

   A list of measures that are candidates for allocation appears.

3. In the **Measure list** box, right-click the **Allocation type** for the measure and select an option.

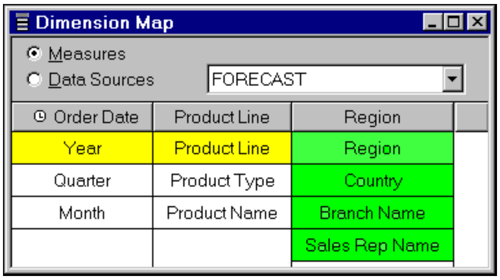
   The allocation types include **From Level**, **Do Not Allocate**, **Constant**, and **By Measure**.

Suppress Allocation

Although allocation is automatic, you can disable it if it is not appropriate for your situation. For example, if the measure has no relationship to a dimension, you do not want it to be allocated as a constant throughout (the default behavior). Similarly, there may be cases where you do not want categories in the next lower level to inherit the allocation type from the higher level.

When you disable allocation, Transformer creates data points with missing values, which appear as zeros in your OLAP reports. You can change this default display to **na** instead.

For example, suppose you track sales revenue each day, for both products and regions, but your sales forecasts are done annually, by product line. By default, Transformer allocates forecast values as a constant throughout the entire Region dimension.

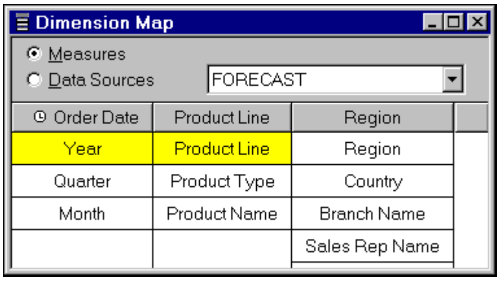


● Measures
○ Data Sources
FORECAST

Order Date	Product Line	Region
Year	Product Line	Region
Quarter	Product Type	Country
Month	Product Name	Branch Name
		Sales Rep Name

However, you do not want the forecast sales appearing in that dimension, as they will not be apportioned equally throughout your regions. On the **Dimension** property sheet, you disable allocation.

You check the scope map to confirm your result.



● Measures
○ Data Sources
FORECAST

Order Date	Product Line	Region
Year	Product Line	Region
Quarter	Product Type	Country
Month	Product Name	Branch Name
		Sales Rep Name

Steps

1. Open the **Dimension**, **Level**, or **Category** property sheet for which you want to suppress default allocation.

2. On the **Allocation** tab, right-click the **Allocation type** and select **Do Not Allocate**.

Setting Up Currency Conversion

If your report users need to see their monetary measures in more than one currency, you can automate the conversion process by adding one or more data sources to your model to supply the rates, and by specifying the necessary conversion details in a built-in **Currency Table**.

Although currency information is stored and applied by default at the lowest level of detail in the time dimension, typically at the Month level, you can change the **Date level** property in the **Currency Record** dialog box to a level that better meets the needs of your users.

Depending on their OLAP reporting component, your users can select the appropriate currency option, and have the conversion rate for that time period applied at run time to the measures in the cube. Converted values are subsequently rolled up using the method you specify in the **Measure** property sheet.

In previous Transformer releases, the currency format specified in the **Currency Table** overrode all other currency format settings. In Transformer version 8.x, if individual measures have a currency format specified, this format overrides the format specified in the **Currency Table**.

The procedure you follow to set up currency conversion depends on your situation. For simple models, you may want to manually create and update your base currency information. For more information, see "Update a Currency Table Manually" (p. 148).

For data that is routinely converted to other currencies, you may find it easier to use external data sources to automatically supply the necessary conversion rates. For more information, see "Enable Conversion Using a Base Currency Table" (p. 143).

If you have legacy data that you must convert to or from the currencies of European Monetary Union (EMU) countries, there may be extra steps (such as euro triangulation) to follow. For more information, see "Enable Conversion Using a Euro Table" (p. 145).

We recommend that you always build cubes in Transformer using the system locale that matches the locale of the measures in your cube, and that you explicitly define a default currency for the data in each model.

Also, if you use more than one data source to supply the conversion rates for the **Currency Table**, you must ensure that the column names match so that the information is combined properly in Transformer.

Enable Conversion Using a Base Currency Table

You use the base **Currency Table** to confirm or change the default currency for the data in your model. Initially, it shows the default currency of the country specified in the locale setting of your operating system, such as **<Base default>** (USA). You can change this to any other currency, including the euro. When you load a conversion table into a populated model, the data appears in the **Conversion Rates** box.

Constraints

You should clear the **Timing** options in the property sheets for all currency data sources so that updates to the **Currency Table** do not interfere with category generation in the rest of your model. If you decide to leave all **Timing** options at their default settings, ensure that the names of the date columns for the base and euro currencies differ from the name of the column used to create your time dimension, and from each other. Otherwise, Transformer cannot differentiate between them.

The **Check Model** tool issues a warning if your model contains a **Currency Table** but none of the measures have currency conversion enabled. You can ignore this message; however, currency conversion will not be available in any OLAP reporting components.

The **Currency symbol** and **Decimals** options available in the reporting components that support run-time currency conversion are determined by the **Country code** list. To override these defaults, you must specify alternatives in the **Currency Record** dialog box.

Steps to Prepare Rate Conversion Source Files

1. Create one or more source files to populate the required currency conversion tables in your model. These files should contain columns for

   - the dates to which the conversion rates apply

   - the codes for the countries to which the conversion rates apply

   - the conversion rates, expressed as the amount by which you multiply a unit of the given currency to convert it to a unit of the base currency

   - optionally, a label for the currency

     If you do not specify a label, it is derived from the country code.

2. If you have a time dimension that does not contain unique levels, you must manually make the levels unique. For more information about uniqueness, see "Recommendation - Resolve Uniqueness Problems in Your Data Source" (p. 46).

3. If categories have not been generated for the time dimension in your model, click **Generate Categories** from the **Run** menu. By generating categories immediately, you do not have to wait until cube creation to see the **Period** information in the **Conversion Rates** box of the **Currency Table** dialog box.

4. Add a data source to your model for each conversion rate source file. If you create the time dimension from more than one source file, make sure that the date column names match.
5. For each source in the **Data Sources** list, double-click it to open its property sheet, click the **General** tab, and clear the **Sets the current period** check box and all check boxes under **Timing**.

   Alternatively, ensure that the conversion date column has a different name from that of the column used to create the time dimension.
6. Click **OK**.

Steps to Create a Base Table

1. From the **File** menu, click **Currency Table** and select the **Use an external currency data source** check box.
2. If you want to change the base currency from the default, which matches the locale setting of your operating system, under **Base Currency**, click **Properties** to open the **Currency Record** dialog box, select a different **Country code** from the list and click **OK**.
3. Confirm that the **Base table columns** box shows the correct names for the **Date**, **Country code**, **Rate**, and optional **Label** source columns. If not, click the ellipsis (**...**) button to the right of the column name box and make corrections in the **New Association** dialog box.

   If the base table columns are not yet defined, click **Add** to specify the associations for mandatory source columns in the **New Association** dialog box.
4. When all of the **Currency Table** options are set, click **Load Table**.

   Conversion rates are added for each period (date category) you specified, and the required currency records are created for your model.

Steps to Enable Currency Conversion

1. Specify the time dimension level to which the conversion rates apply by clicking each currency in the **Currencies** box of the base **Currency Table** and clicking **Properties** to open the **Currency Record** dialog box.
2. Confirm that the **Date level** property is correctly set.
3. Click **OK** to update the currency data.
4. Open the property sheet for each measure that requires currency conversion and, on the **General** tab, click **Allow currency conversion**.

Example - Converting Measures Using a Base Currency File

You use a comma-delimited text file to supply currency conversion rates for the periods defined by the time dimension in your model.

This excerpt shows rates for the final quarter of 2006, including three mandatory columns for the **Currency Record** dialog box (**Date**, **Country code**, and **Rate**), and the optional column **Label**.

To start, conversion rates have a default value of `1.0000`, but you can change the value for any period that is marked with the pencil icon ✏. Note that all rates represent the amount by which you multiply a unit of the given currency to convert it to a unit of the base currency. For this example, in December 2006, 1 US dollar (base currency) cost 1.10 Canadian dollars, 0.60 British pounds, and 112.90 Japanese yen.

```
DATE,LABEL,COUNTRY CODE,CONVERSION RATE
20061001,Canadian Dollars,CAN,1.1011
20061101,Canadian Dollars,CAN,1.1084
20061201,Canadian Dollars,CAN,1.1046
20061001,British Pound,GBR,0.6022
20061101,British Pound,GBR,0.6057
20061201,British Pound,GBR,0.6042
20061001,Japanese Yen,JPN,113.4700
20061101,Japanese Yen,JPN,113.1863
20061201,Japanese Yen,JPN,112.9033
```

Enable Conversion Using a Euro Table

If you have legacy (turn-of-the-century) data and select the **Use Euro triangulation** option, you can convert values to or from the currencies of European Monetary Union (EMU) countries, using the method stipulated for the transition period.

For those few years when EMU-member countries carried two legal tenders, special rules apply. Rates had to be expressed in terms of the euro, and values had to be converted from the original currency to the euro, and then from the euro to the target currency, with rounding to the specified precision at each step.

Additional information to be specified in the **Currency Record** dialog box includes **Entry date in the EMU (YYYYMMDD)** and the fixed rates to use when converting data to or from the national currencies of the EMU countries. As with the base **Currency Table**, rates can be added to the euro **Currency Table** for the relevant time periods, either manually or by means of one or more external data sources.

An **Entry date in the EMU (YYYYMMDD)** is automatically specified for the euro, which is the default currency in the euro **Currency Table**. Conversion rates created for the **<Base default>** currency before that date are set to zero. This means they appear as **NA** in Transformer, as zero in Model Definition Language (MDL), and as missing or **na** in the reporting components.

If you use euro triangulation, the date column in your data source must be sorted in ascending order and all date levels must be set to **Unique** on their property sheets. Remember that you cannot change externally supplied rates after they have been imported into your **Currency Table**. Fixed rates are those not marked with the pencil icon ✏.

The **Check Model** tools issues a warning if your model contains a **Currency Table** but none of the measures have currency conversion enabled. You can ignore this message; however, currency conversion will not be available in any OLAP reporting components.

You should clear the **Timing** options in the property sheets for all currency data sources so that updates to the **Currency Table** do not interfere with category generation in the rest of your model. If you decide to leave all **Timing** options at their default settings, ensure that the names of the date columns for the base and euro currencies differ from the name of the column used to create your time dimension, and from each other. Otherwise, Transformer cannot differentiate between them.

The **Currency symbol** and **Decimals** options available in the reporting components that support run time currency conversion are determined by the **Country code** list. To override these defaults, you must specify alternatives in the **Currency Record** dialog box or on the **Format** tab of the **Measure** property sheet.

Steps to Prepare Euro Conversion Source Files

1. If you have not already done so, create and load the required source files for your base **Currency Table** source files, and then create the corresponding source files to populate the euro portion of your currency conversion table. These files should contain the three mandatory columns, and may contain a fourth currency **Label** column, as described below:

   - For each EMU-member country, insert a zero value for the date. For countries that are not EMU members, add a row for each time period to which the conversion rates apply, and insert the correct date. This constitutes the **Date** column.

   - Uniquely identify the countries to which the rates apply, using the internationally recognized code for each country. If the country already appears in the base **Currency Table**, the code must be identical to that used there. This constitutes the **Country Code** column.

   - Specify the amount by which you must multiply one unit of the currency to convert it to a euro. This constitutes the euro conversion rate column.

   - Optionally, specify a meaningful label for the currency. If you do not, the label is generated automatically, from the country code.

2. Add a data source to your model for each euro conversion source file.

   Note: If you create the time dimension from more than one source file, the **Date** column names must match. Also, the column names specified for euro conversion must differ from those specified for the base **Currency Table**.

3. Double-click each source in the **Data Sources** list to open its property sheet, click the **General** tab, and clear the **Sets the current period** check box and all check boxes under **Timing**.

   Alternatively, ensure that the conversion **Date** column has a different name from that of the column used to create the time dimension.

4. Click **OK**.

Steps to Create a Euro Table

1. From the **File** menu, click **Currency Table** and select the **Use Euro triangulation** check box.

2. Confirm that the **Base table columns** box shows the correct names for the **Date**, **Country code**, **Rate**, and optional **Label** source columns, to populate each currency in the euro **Currency Table**. If not, click the ellipsis (**...**) button to the right of the column name and make corrections in the **New Association** dialog box.

   If the euro table columns are not yet defined, click **Add** to specify the associations for mandatory source columns in the **New Association** dialog box.

3. For each country that is an EMU member and needs to appear in your euro **Currency Table**, perform the following steps:
    - Right-click the country in the **Currrencies** box and click **Properties** to open the **Currency Record** dialog box.
    - Select the **Member of the Economic and Monetary Union** (**EMU**) check box.
    - Accept the default **Entry date in the EMU** (**YYYYMMDD**) (**19990101**) or type a different date and click **OK**.
4. When the euro **Currency Table** is populated with the fixed rates for EMU countries and any non-EMU **<Base default>** countries, and all other **Currency Table** options are properly set, click **Load Table**.

    Conversion rates are added for each period (date category) you specified, and the required currency records are created for your model.

Steps to Enable Currency Conversion

1. Specify the time dimension level to which the conversion rates apply by clicking each currency in the **Currencies** box of the base **Currency Table** and clicking **Properties** to open the **Currency Record** dialog box.
2. Confirm that the **Date level** property is correctly set.
3. Click **OK** to update the currency data.
4. Open the property sheet for each measure that requires currency conversion and, on the **General** tab, click **Allow currency conversion**.

Example - Converting Legacy Data Using Euro and Base Currency Files

You use two comma-delimited text files to supply euro and base currency conversion data for the euro transition periods encompassed by the time dimension in your model.

The following excerpts (Q1 1999) are for illustration purposes only. To obtain actual conversion rates for euro triangulation, please visit European Commission or EMU Web sites.

```
DATE,LABEL,COUNTRY CODE,CONVERSION RATE
19990101,Canadian Dollars,CAN,1.5223
19990201,Canadian Dollars,CAN,1.5184
19990301,Canadian Dollars,CAN,1.5146
19990101,British Pound,GBR,0.6072
19990201,British Pound,GBR,0.6057
19990301,British Pound,GBR,0.6042
19990101,Japanese Yen,JPN,113.4700
19990201,Japanese Yen,JPN,113.1863
19990301,Japanese Yen,JPN,112.9033
```

Sample euro data follows. Note the different column names and the use of zero for the date.

```
EURO DATE,EURO LABEL,EURO COUNTRY CODE,EURO
CONVERSION RATE
0,Austrian Schilling,AUT,13.7603
0,Belgian Franc,BEL,40.3399
0,Finnish Markka,FIN,5.94573
0,French Franc,FRA,6.55957
0,German Deutsche Mark,DEU,1.95583
```

```
0,Italian Lira,ITA,1936.27
0,Dutch Guilder,NLD,2.20371
0,Spanish Peseta,ESP,166.386
19990101,U.S. Dollar,USA,1.16741
19990201,U.S. Dollar,USA,1.17028
19990301,U.S. Dollar,USA,1.17330
```

Conversion rates are expressed as the amount by which you multiply a unit of the given currency to convert it to a unit of the base currency. For example, in March 1999, 1 US dollar (base currency) cost 1.51 Canadian dollars, 0.60 British pounds, and 112.90 Japanese yen. On the same date, 1 euro cost 6.56 French francs (fixed on entry into the EMU) or 1.17 US dollars (floating exchange rate).

Update a Currency Table Manually

If your model design is simple and does not use external data sources to supply conversion rates, you can manually create and update the required currency tables.

If you used an external data source to create a preliminary **Currency Table** but decide to maintain it manually, remember to clear the **Use an external currency data source** check box.

Steps

1. From the **File** menu, click **Currency Table**.
2. In the base **Currency Table**, click the default currency in the **Currencies** box and click **Add New Currency**.
3. Select the required **Country code**, set other properties as required, and click **OK**.
4. In the **Conversion Rates** table, click the **Conversion Rate** you want to change, and make the required updates.
5. Repeat step 4 for every rate that you want to change.
6. Click **OK** to include your new conversion rate data in the model.

Chapter 6: Creating PowerCubes

When you select **Create PowerCubes** from the **Run** menu, one or more PowerCubes are generated with the default property settings. These binary files contain all the source data for the dimensions and levels in your model.

PowerCubes generated by Transformer version 8.x can be used as a data source for any IBM Cognos 8 version 8.3 Web studio, such as Analysis Studio, Report Studio, or Metric Studio. If the PowerCube does not have IBM Cognos 8 security, it can be used in IBM Cognos Series 7 PowerPlay products, and in IBM® Cognos® 8 Business Intelligence Mobile Analysis for local (disconnected) use. Once IBM Cognos 8 security is included, the PowerCube can only be used in IBM Cognos 8 version 8.3 Web studios.

However, before you deploy your cubes, you may want to customize them. For example, you can create cube groups where the child cubes are intended for different users, and so contain different dimension views or subsets of the query objects defined in your model. You can also improve performance by setting up time-based partitioned cubes. For more information about time-based partitioning, see "Defining a Time-based Partitioned Cube" (p. 151).

When you are designing your cubes, you should evaluate the languages in which your end users will require the cubes to be distributed. For information about cube languages and locales, see "Managing Languages and Locales" (p. 185).

Tip: We recommend that you experiment until you have a satisfactory balance between cube build and data retrieval times. This customization may, but need not, parallel the security-based views that you associate with your configured IBM Cognos namespace.

You can create

- ❑ similar but different models that meet the needs of each target group, perhaps pointing to different source files

  You can then build one focused cube from each of these models.

- ❑ one model and one cube, and use dimension views to remove unwanted categories and offer different perspectives on the data

- ❑ a group of several smaller, related cubes from one model, and control the measures and details created in each cube

- ❑ a large time-based partitioned cube and several smaller cubes, related across a conformed time dimension

- ❑ one large shared cube, and hide certain dimensions from some users by implementing IBM Cognos 8 custom views

  For more information about custom views, see "Adding Security" (p. 173).

Create a Single PowerCube

When you want to tailor the properties of a PowerCube to meet the specific needs of users, you can manually define the cube. Examples of items that you can customize include the location of the PowerCube or the dimensions and measures to include.

Each cube uses Structured Query Language (SQL) to retrieve the specified query items and their relationships from the data sources in your model. When you create or modify a cube, you can also apply security to it, to control your users' access to information.

For example, if your data source contains information from different areas of a business, you might decide to create different cubes for Human Resources and Finance. You can define each cube to meet a set of broadly defined, but related, OLAP reporting needs. Each report contains only the information from a single cube, although you can support additional analysis by setting up drill through to related reports. For more information, see the topic on drill-through access in the IBM Cognos 8 *Administration and Security Guide*.

To create a cube, your model must have at least one dimension with at least one level, and its **Measures** list must show at least one measure.

Steps

1. Click in the **PowerCubes** list to make it active and, from the **Edit** menu, click **Insert PowerCube**.

2. In the **PowerCube name** box, type a name for the new PowerCube.

3. In the **PowerCube file name** box, click **Browse** to specify the location where the PowerCube will be saved.

   When you specify the cube name on the **Output** tab, ensure the path is valid and avoid using restricted characters, such as the exclamation mark (!), in the file name.

   By default, Transformer saves cubes in the My Documents/Transformer/PowerCubes directory. You can set the default location to which Transformer saves models by changing the **PowerCubes** directory setting on the **Directories** tab of the **Preferences** property sheet. On Windows Vista, Transformer saves cubes in the Documents/Transformer/PowerCubes directory.

   We recommend that you specify cube names without spaces in the file name. Although spaces are valid, file names without spaces are preferable for UNIX/Linux environments.

4. To save a cube that is in use and therefore locked, select the **Use temporary file name if the original file is locked** option and specify an alternative name for the PowerCube file.

5. On each of the other tabs in the **Powercube** property sheet, set the properties you want to apply.

6. When you have finished defining your PowerCube, click **OK**.

7. To review your initial model design and confirm the results of your cube build, open the cube in PowerPlay: select the cube and then click the **PowerPlay for Windows** button.

   To open a cube in PowerPlay version 7.x, you must not have applied any custom views to the cube.

If you receive an error message indicating that the setting for the PowerPlay application is either not defined, or is not valid, click the **Browse** button to locate the PowerPlay application. The PowerPlay application location is defined on the **Directories** tab of the **Preferences** property sheet.

Create a Cube Group

You can create a cube group for any level in a dimension and add child cubes that include selected categories or measures. For each cube in a group, you must specify the degree of detail to be included so that your users see only the most relevant data.

For example, suppose you work for a national chain of travel agencies that uses one data source and one model to generate its sales reports. You create one large cube that can be accessed by several different user groups and roles, and a set of smaller cubes that are each accessed by only one user group. Senior managers see total and regional sales figures for each travel product. Regional managers see figures for only their regions.

Although it takes longer to build several smaller cubes, managers will see faster query response times, and each regional cube contains data relevant to only that regional manager.

Steps

1. Click in the **PowerCubes** list to make it active and, from the **Edit** menu, click **Insert PowerCube**.
2. In the **PowerCube name** box, type a name for the new PowerCube group.
3. Click the **Cube Group** tab.
4. In the **Dimension** box, select the dimension from which you want to build the cube group.
5. In the **Level** box, select the level whose categories will become the individual cubes in the cube group.
6. In the **Focus of detail** area, select the lowest level of category detail.

   Note: This is the level at which you want data for external categories to be summarized.
7. When you have finished setting the required properties on each of the other tabs in the **PowerCube** property sheet, click **OK**.

Defining a Time-based Partitioned Cube

Time-based partitioning is a useful means of optimizing cubes for OLAP reporting purposes. Time-based partitioned cubes are a collection of child cubes, based on one level in the time dimension, that together form one large cube. Each child cube is partitioned, or split, at the appropriate reporting level, such as Quarter or Month.

Report users can view each cube independently, or access the entire collection of child cubes as a single, time-based virtual cube. This means that reports can be viewed across the entire time dimension, or across only one level in the time dimension, such as a specific month.

Transformer creates and maintains two separate files to manage the time-based cube group. The first is a time-based control file (.vcd), which is an editable ASCII-format text file that references the cubes that make up the time-based group and their physical locations. The second file is a control cube (.mdc), which:

- contains the high-level metadata about the overall structure of the cubes
- maintains a list of all measures, dimensions and root categories, currency records, and the entire structure of the time dimension
- contains the control information to gather the data from the child cubes
- serves as the entry point for OLAP report users to access the time-based partitioned cube

The .mdc file shares the same file name and location as the .vcd file.

For example, suppose your sales organization can benefit from a time-based partitioned cube group defined on the Quarter level, so cubes can be rebuilt every quarter.

After one year, you have separate cubes for the four quarters of 2006. In the first quarter of 2007, you run an update creating a 2007_Q1 child cube. The large control cube has categories in the time dimension for 2006 and 2007. The text file (.vcd) generates five lines, one for every quarter from 2006_Q1 to 2007_Q1.

Advantages of a Time-based Partitioned Cube

Time-based partitioned cubes are similar, but superior, to standard cube groups, in that they

- offer a faster, more efficient way of building and updating time-segmented data than incrementally updated cubes

  New data is typically added to a single partitioned cube, rather than to a large existing cube.

- eliminate the periodic need to do a full build, resulting in shorter down-times for the production system, and a more easily managed maintenance schedule
- support rolling time periods

  You can manually edit the .vcd file to remove references to cubes that are no longer required, and drop invalid or out-of-date categories. The control cube and definition file are automatically updated with the newest categories and cube references. For more information, see "Customizing a Time-based Partitioned Cube" (p. 157).

- support slowly changing dimensions

  Existing child cubes retain the history, and new cubes are easily created using the **Move** capability for categories.

- offer better query performance, because users drilling down into the time dimension encounter fewer cubes

  When report users reach the level of granularity that the cubes are based on, such as January or Q1, they only need to access a single cube.

- offer more flexibility

Although time-based partitioned cubes relate to only one level of granularity, such as Month, you can still reference other time levels in the same model or cube if this improves run-time performance.

Disadvantages of a Time-based Partitioned Cube

Category handling, sorting enhancements, and support for external rollup make time-based partitioning a useful optimization technique for production environments.

However, the following restrictions apply:

- Your model can only have one time dimension.

  If the model has more than one time dimension, the **Enable time-based partitioning** check box is permanently disabled.

- Dimension calculation definitions that create calculated categories are not supported.

  You cannot use a calculated category at the time dimension level in a time-based partitioned cube.

- Special categories created manually are supported, but the placement of these special categories in a time-based partitioned cube depends on the type. For example,

  - Time-related special categories cannot be located in the child cubes, but are available from the control cube.

  - Regular special categories appear in the child cubes when they are initially built, but new special categories cannot be added to child cubes during subsequent updates.

- New regular categories may be added to the child cubes but, if they are, special categories or links to special categories are not created.

- Category counts are supported in the primary drill-down path only if the counted level is unique and an **Activity measure** is explicitly set on the **Rollup** tab of the **Measure** property sheet.

  If this setting is left as the default (all measures in the cube), at run time, some measures will be disregarded and the count will not accurately reflect all categories with non-zero values.

  Also, the **Activity measure** for the category count rollup cannot be one of the following:

  - a before or after-rollup calculated measure

  - an externally rolled-up measure

  - another category count measure

  - an allocated measure, if the allocation occurs in the same dimension as the category count, above the counted categories level

- Category inclusion is based on the time level specified when the time-based partitioned cube was first created.

  Not all children of a category are included in each cube. Only those categories that contain data for the given time period appear in the resulting reports.

- Support for external rollup is as follows: if the child cube has a custom view based on the time partition level, and this custom view excludes or cloaks categories of that level, you can open the child cube. However, the excluded categories are not shown and the value of the time dimension root is returned as missing (N/A or zero).
- Allocations that are performed from a level above the time-based partitioning level are not supported. Categories can only be allocated below the time-based partitioning level.
- Temporary cube filenames are not supported because their use can yield incorrect results.
- Drill through is supported. However, unexpected or inconsistent results may arise if, for example, users drill through from a time-based partitioned cube containing data that is categorized differently among the child cubes in the group.
- Child cubes are always published with the time-based cube. Disabling publishing for a child cube only restricts publishing of that cube individually.

Create a Time-based Partitioned Cube Group

To more easily manage time-based partitioned cubes, you can gather them into like-structured groups.

Steps

1. To set up the cube group, on the **PowerCube** property sheet, click the **Cube Group** tab, select the **Enable time-based partitioning** check box, and click **OK**.
2. To ensure that the child cubes in your group cover a distinct level, for each one, specify the appropriate level from your time dimension, such as Quarter or Month.

   Tip: You can open the .vcd file later, in any text editor, and manually include or exclude cubes. For example, to improve performance, try adding entries in the .vcd file for cubes that are at a higher level of the time hierarchy, such as the Quarter or Year level. For more information, see "Customizing a Time-based Partitioned Cube" (p. 157).

Processing Issues Related to Time-based Partitioned Cubes

Time-based partitioning can change how categories are sorted, depending on which categories are included in a particular partition.

Every child category does not appear in every child cube. Category inclusion is based on the time level specified when you enable partitioning. Data outside the specified time period is not included in the time-based partitioned cube.

Your goal as modeler is to define the partitioning so that every child cube contains all the relevant or possible categories; that is, an inclusive approach. However, this may not be feasible if, for example, all new categories are added only to the latest cube. In such cases, the sort order for the categories is stored in the individual child cubes, called disjoint cubes, and not in the large cube.

Sorting Considerations

If you make any of the following changes to your model, unexpected sorting may result:

- multiple category deletions
- the addition of many new categories that lack contextual information about their parent-child and child-sibling relationships
- use of the **Unique** and **Move** options for categories in a level

Even when the control cube contains high-level metadata about the overall structure of the cubes, frequent changes to the metadata may affect how time-based partitioned cubes perform. For sorting to work properly, the changes to the categories must be relatively slow and orderly, and there must be no significant changes to the metadata categories in the overall model.

Because time-based partitioned cubes handle large amounts of time-related OLAP data in a seamless, efficient manner, cube updates and query times are normally faster. However, querying may be slower due to the extra time needed to sort across multiple child cubes, and there may also be an increased memory requirement. This performance overhead varies, depending on such factors as the number of cubes referenced by the query and the number of child categories in each parent being sorted.

Adding New Data to Time-based Partitioned Cubes

New categories are added to the newer cubes in the normal manner. However, older, pre-existing categories may not have new data associated with them in the more recent cubes. In such cases, not every child cube may contain the complete set of metadata; that is, the category information, relationships, and measure values.

If the category lists do not remain stable over time, or if major changes occur between cube updates, it may be difficult for Transformer to understand and correctly interpret the context of any moved categories. That is, the integrity of the category hierarchy, or parent-child and sibling relationships, may not be reflected in each child cube.

Use of the **Unique** and **Move** options further complicates the situation, because the moved category becomes separated from the parent and siblings that indicate its context. This can produce unexpected sorting results.

For example, suppose your model design includes several child cubes, and the data added to your time-based partitioned cube includes new categories that did not previously exist. Because there is an overlap in the time periods, the overall cube sort order reflects a merging of the categories. More recent categories are appended to the end of the list, destroying the sort order. The parent context is unavailable in the child cubes, so the category list cannot include categories missing from the child cubes. If your query spans multiple child cubes, the overall sort order is a merging of the categories from the individual cubes.

Example - Sorting After the Addition of Categories to Disjoint Child Cubes

Suppose your model design includes disjoint child cubes. The overall category order in the parent cube is A, B, C, D. Categories A and B appear in one child cube; C and D appear in the other.

You add a new category, E, to the A+B child cube. A special algorithm correctly sets a sort order in the reporting component to A, B, C, D, E, recognizing that E is the child of the same parent as A, B, C, and D.

Note: Time-based partitioned cubes are optimized to undergo incremental updates in which the metadata changes gradually over time. They are not designed for cases where significant numbers of child categories are deleted or moved, using **Move** option on the **Level** property sheet, for example. Operations such as these remove historical context, producing disjoint child cubes in which unexpected sort orders may appear.

Slowly Changing Dimensions

In your OLAP reporting components, a group of time-based partitioned cubes appears as a single virtual cube, but the data in the individual cubes represents business information that may be changing slowly over time.

Suppose that, over several months, you need to delete a category, add a new category in its place, and move the added category to a new parent in the same dimension. Transformer can maintain the required category code identifications because the changes are sufficiently gradual, from one update to the next. Because the parent-child context is preserved, the original sort order can be successfully maintained.

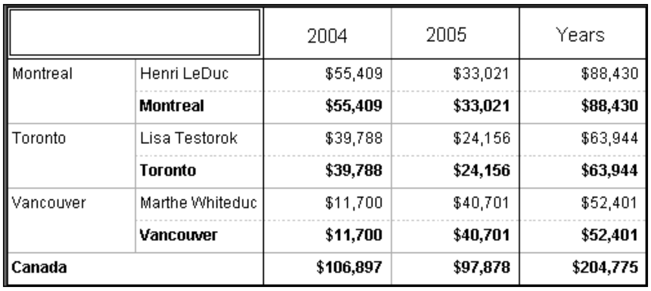
Considering the warnings about unexpected sort order, you can use unique moves in time-based partitioned cubes by selecting both **Unique** and **Move** on the **Level** property sheet. Any changes in the model are reflected in the new child cubes.

If you make a change to the model and then build your time-based partitioned cube group, a warning appears indicating that the model has changed. If you continue with the build, the new child cube is built reflecting the changes.

Note: Historical information continues to exist in older child cubes; new information is placed in the new child cubes. It is possible that rollups will not reflect the expected values because they may be using data from two sources.

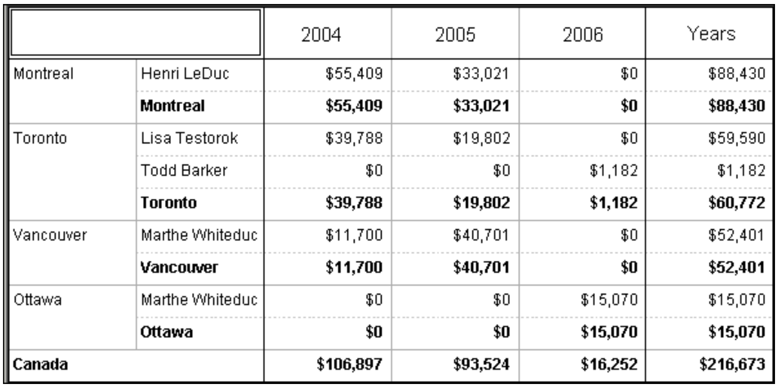
Example - Sorting in a Slowly Changing Dimension

The following report shows sales by the sales representative Marthe for 2004 and 2005.



		2004	2005	Years
Montreal	Henri LeDuc	$55,409	$33,021	$88,430
	Montreal	**$55,409**	**$33,021**	**$88,430**
Toronto	Lisa Testorok	$39,788	$24,156	$63,944
	Toronto	**$39,788**	**$24,156**	**$63,944**
Vancouver	Marthe Whiteduc	$11,700	$40,701	$52,401
	Vancouver	**$11,700**	**$40,701**	**$52,401**
Canada		**$106,897**	**$97,878**	**$204,775**

When Marthe moved from Vancouver to Ottawa in 2006, the report displays the historical data for 2004 and 2005 in Vancouver, and the new data for 2006 in Ottawa.



		2004	2005	2006	Years
Montreal	Henri LeDuc	$55,409	$33,021	$0	$88,430
	Montreal	**$55,409**	**$33,021**	**$0**	**$88,430**
Toronto	Lisa Testorok	$39,788	$19,802	$0	$59,590
	Todd Barker	$0	$0	$1,182	$1,182
	Toronto	**$39,788**	**$19,802**	**$1,182**	**$60,772**
Vancouver	Marthe Whiteduc	$11,700	$40,701	$0	$52,401
	Vancouver	**$11,700**	**$40,701**	**$0**	**$52,401**
Ottawa	Marthe Whiteduc	$0	$0	$15,070	$15,070
	Ottawa	**$0**	**$0**	**$15,070**	**$15,070**
Canada		**$106,897**	**$93,524**	**$16,252**	**$216,673**

In other words, after the model structure changed, the older child cubes were locked and were therefore not updated with the new data. The control cube combined data stored in the newest child cube and therefore could update the parent value in the time-based partitioned cube.

Customizing a Time-based Partitioned Cube

You can customize a time-based partitioned cube to enhance performance by manually editing the .vcd file in any text editor. This ASCII file resides in the same directory as the control cube, under the same file name, and contains one line for each child cube in the group.

For example, you can remove child cubes in the following .vcd file, to exclude one or more of the eight cubes defined at the Quarter level from the time-based partitioned group:

```
cube "2006 Q1" .\PowerCube\2006 Q1.mdc
cube "2006 Q2" .\PowerCube\2006 Q2.mdc
cube "2006 Q3" .\PowerCube\2006 Q3.mdc
cube "2006 Q4" .\PowerCube\2006 Q4.mdc
cube "2007 Q1" .\PowerCube\2007 Q1.mdc
cube "2007 Q2" .\PowerCube\2007 Q2.mdc
cube "2007 Q3" .\PowerCube\2007 Q3.mdc
cube "2007 Q4" .\PowerCube\2007 Q4.mdc
```

Note: The paths defined in the .vcd file are relative to the location of the large control cube, as indicated by the leading dot: `cube "2006 Q1" .\Cube\2006 Q1.mdc`. You can also specify a full path to each child cube: `cube "2006 Q1" C:\Transformer\Cube\2006 Q1.mdc`.

You can also create a time-based partitioned cube that is based on several different levels of the time dimension. For example, to improve run-time performance, your users might prefer to query against a cube that is based on a year cube, three quarter cubes, and a month cube.

To add these time-based partitioned cubes, you must first generate the child cubes for the specific time levels and then edit the .vcd file so that it contains only the required child cube references. When you have edited the .vcd file, the five remaining cubes in the .vcd file become accessible through the control cube:

```
cube "2006" .\Cube\2006.mdc
cube "2007 Q1" .\Cube\2007 Q1.mdc
cube "2007 Q2" .\Cube\2007 Q2.mdc
cube "2007 Q3" .\Cube\2007 Q3.mdc
cube "October" .\Cube\October.mdc
```

Exclude Measures from a Cube

You can customize a PowerCube or cube group by excluding certain measures so that they cannot be viewed in the OLAP reporting components. You can remove them entirely, or you can hide them from specific user groups defined in your configured IBM Cognos namespace. In such cases, all measure values are retained in the cube, but the restricted dimensions cannot be accessed by the excluded users.

For example, suppose your cube summarizes your sales data for sporting goods. In addition to 2006 sales, there are measures for product cost and 2005 sales, which your users do not need. You decide to exclude these last two measures from the cube.

Tip: Be careful not to exclude measures that are used in the calculation of other measures used in the model. Otherwise, errors may occur when your cube is created or updated.

Steps

1. On the **PowerCube** property sheet, click the **Measures** tab.
2. Right-click a measure and click **Exclude**.
3. Click **OK**.

Omit Dimensions from a Cube

You can omit specific dimensions from a cube to customize it for particular users. You can omit a dimension from the cube entirely, or you can include it but hide it from specific users.

For example, suppose your model uses several data sources to provide information about every facet of your sporting goods business. The source columns are Product, Customer, Branch, and Order, with each having a corresponding dimension. You want one of your cubes to focus on product sales performance. Because you do not need customer data, you omit that dimension from the cube.

For models with authentication enabled, you must use the **Categories** diagram to hide a dimension from a user group, rather than omit it from the cube. In such cases, the dimension is retained in the cube but restricted users cannot access it.

Steps

1. On the **PowerCube** property sheet, click the **Dimensions** tab.
2. Right-click a dimension to omit or hide and click **Omit Dimension**.
3. Click **OK**.

Customizing Cube Content with Views

Instead of omitting measures or entire dimensions from a cube, you can select one of the following customization options, to deliver only that information of interest to your users:

- **Apex** omits ancestors and siblings of a category.

Use the **Apex** option with dimension or custom views.

- **Exclude** omits a category, its descendants, and their data.

  Apply **Exclude** from the category viewer and use this option with dimension or custom views.

- **Cloak** omits a category and its descendants, but retains the rollup values in ancestor categories.

  Use the **Cloak** option with dimension or custom views.

- **Summarize** omits descendants, but retains their rollup values.

  Use the **Summarize** option with dimension or custom views.

- **Suppress** omits a category from reports based on the cube, but retains its rollup value in ancestor categories.

  Apply **Suppress** from the category viewer or with dimension views. For special categories, **Suppress** is the only available option.

Note: The **Cloak**, **Summarize**, and **Suppress** options do not remove data from the cube. Therefore, even if many categories are omitted, there is no corresponding decrease in cube size.

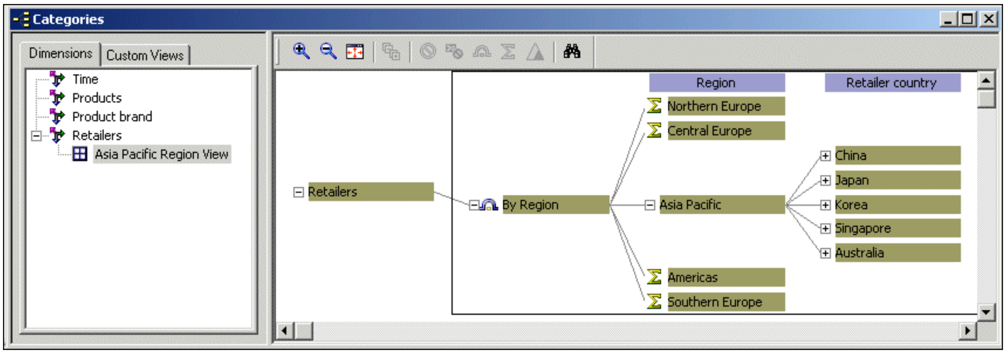
You create dimension and custom views in much the same way, but you must configure security before you include custom views in your model. You can then select security objects from the available list, define a custom view using any option except **Suppress**, and associate the view with a cube in the **PowerCubes** list. When a user opens the cube in PowerPlay or one of the IBM Cognos 8 studios, the system verifies that the user has the necessary access permissions for the secured data before showing the portion of the cube defined in the custom view.

You can create cubes for different audiences based on dimension views, and further restrict access by applying custom views to selected cubes in the set. However, certain combinations may not work well with your partitioning scheme, thereby lengthening cube build times. One workaround is to create several smaller cubes, each using different dimension views, rather than create one large cube that has an unacceptably long build time.

Create a Dimension View

When you need to develop OLAP reporting solutions for diverse groups in your organization, you can create dimension views and apply them to the cubes in your model, thereby creating customized cubes and drill-through reports to meet the needs of each group.

For example, suppose you want to create a different cube for each regional sales manager that contains the transaction details for their region and summarized values for the other regions. You begin by defining an Asia Pacific Region View that summarizes the data for Northern, Central, and Southern Europe and the Americas, while retaining the details for the Asia Pacific region.



You apply your dimension view by selecting the **View** option on the **Dimensions** tab of the property sheet for the **PowerCube** object. You then repeat these steps to create the required dimension views in your other regional cubes.

The resulting regional cubes contain detailed data for each manager of that region and summarized information for the other regions.

Tip: Instead of creating a separate dimension view for each category in a level, you may want to create a cube group based on that target level, if this is easier to implement and maintain. Because only a subset of the categories is written to the cube, a cube that implements dimension views often has the advantage of being smaller and therefore quicker to build and to deploy.

You should keep the following constraints in mind when implementing this feature:

- You cannot base your partitions on the dimension used to create a dimension view. If you do, conflicts can result, which in turn may halt the cube creation process.
- You should not use dimension views as the basis for custom views, as this may negatively affect auto-partitioning.

For more information about custom views, see "Adding Security" (p. 173).

Steps

1. Open the **Categories** diagram for the relevant dimension.

   Tip: If the **Dimensions** pane is not showing, click the **Dimensions** tab.

2. In the **Dimension** pane, right-click the dimension and click **Add New View**.

3. In the **View name** box, type a meaningful name for the dimension view and click **OK**.

   You can also type a description of the new dimension view in the **Description** box.

4. In the category viewer, select the required categories and, from the **Diagram** menu, customize the new dimension view by selecting the category actions that will help you achieve the summarization you want.

   For more information, see

   - "Omit Categories Using Suppress" (p. 161).
   - "Omit Categories Using Cloak" (p. 162).
   - "Omit Categories Using Exclude" (p. 163).

- "Omit Descendant Categories Using Summarize" (p. 164).
- "Omit Categories Using Apex" (p. 164).

5. Open the **PowerCube** property sheet and click the **Dimensions** tab.
6. In the **Dimensions** list, right-click the dimension in which you created the dimension view and click **View**.
7. In the **Select a View** dialog box, select the dimension view that you created, and click **OK** twice.

Omit Categories Using Suppress

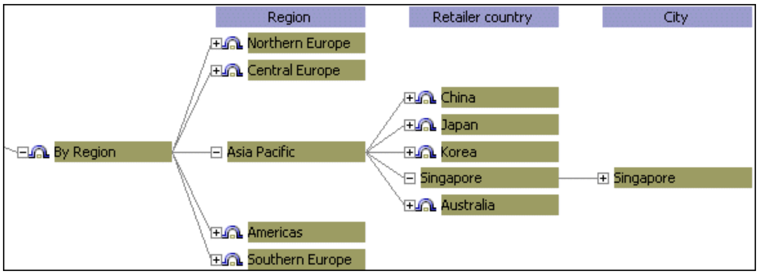
In situations where a category is not required for reporting purposes, but its ancestor and descendant categories are, you can suppress the category to hide it.

Suppression creates placeholder categories in hierarchies where no values exist for some categories in a level. The values for the descendants of a suppressed category are rolled up and retained in the ancestor category.

Suppressed categories, including blank categories, are still included in category counts. Values associated with them are included in rolled-up measure values.

For example, suppose that your company maintains branches throughout the world. Some branches are identified by their country. Others are identified by city only.

In your model, you structure the Regions dimension to show levels for Retailer country and City. To streamline your cube, you suppress categories with no associated source data.



In the resulting report, your users can drill directly down to subordinate categories that reflect the current structure of your organization.

You can suppress both existing and blank categories in a level by suppressing the level itself. However, to properly support the calculation of relative time categories for the cube, you cannot suppress or exclude weekend days (Saturdays and Sundays) and their associated measure values.

You can also suppress categories in different alternate drill-down paths, but you cannot suppress more than one entire alternate drill-down path. Suppressing any category located above the convergence level in a drill-down path has no effect on categories in other drill-down paths.

Steps

1. Open the **Categories** diagram for the dimension that contains the categories or level to be suppressed and, if you have not already done so, add a dimension view.

   For more information, see "Create a Dimension View" (p. 159).

2. In the **Dimension** pane, select your new view.

3. In the category viewer, select the categories or level to be suppressed and, from the **Diagram** menu, click **Suppress**.

4. Open the **PowerCube** property sheet and click the **Dimensions** tab.

5. In the **Dimensions** list, right-click the dimension in which you created the dimension view and click **View**.

6. In the **Select a View** dialog box, select the dimension view that you created, and click **OK** twice.

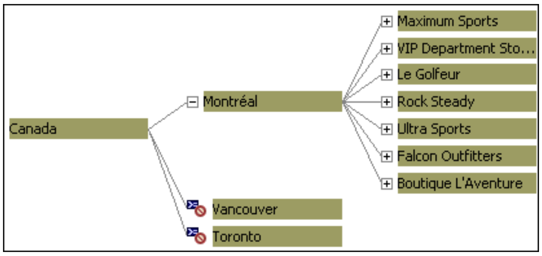
Omit Categories Using Cloak

In certain situations, you may want to omit one or more categories with the descendants of those categories, while retaining their values for rollup into higher-level categories.

Like **Suppress**, the **Cloak** option causes a category to be omitted from reports based on the cube. Unlike **Suppress**, all descendants of that category are also omitted.

When you apply cloaking to a cube, consolidation occurs by default. If new subordinate categories are added to a cloaked level, they are automatically included in the cloaked ancestors.

For example, suppose that the French-speaking manager for Belgium and France also oversees your Montreal office. You define a special dimension view for this manager that includes the transaction details for Europe and Montreal (Canada), but that cloaks the values for the two other Canadian offices, Toronto and Vancouver.



You apply this view to the **PowerCube** object. The resulting cube contains data relevant to this manager.

Steps

1. Open the **Categories** diagram for the dimension that contains the categories or level to be cloaked and, if you have not already done so, add a dimension view.

   For more information, see "Create a Dimension View" (p. 159).

2. In the **Dimension** pane, select your new view.

3. In the category viewer, select the categories or level to be cloaked and, from the **Diagram** menu, click **Cloak**.

4. Open the **PowerCube** property sheet and click the **Dimensions** tab.

5. In the **Dimensions** list, right-click the dimension in which you created the dimension view and click **View**.

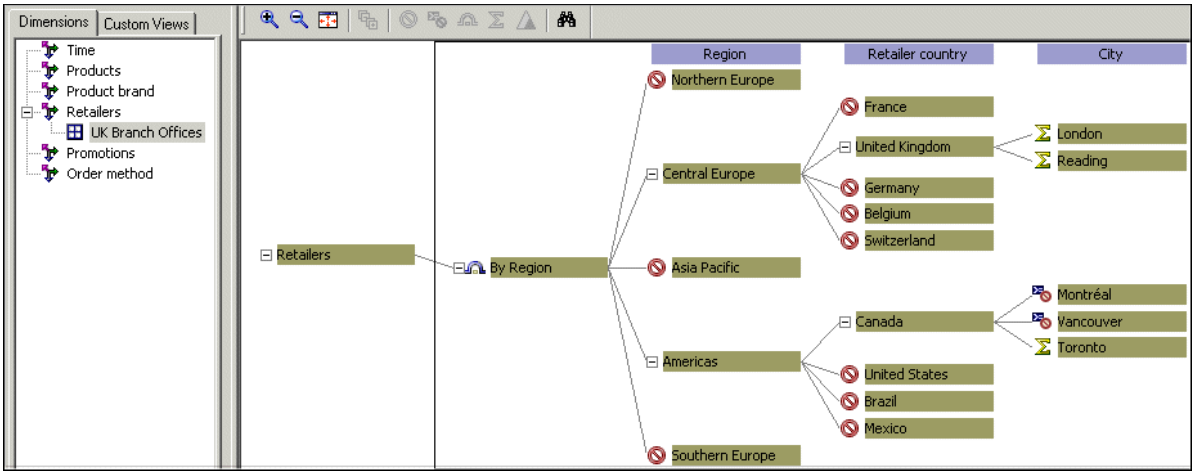
6. In the **Select a View** dialog box, select the dimension view that you created, and click **OK** twice.

Omit Categories Using Exclude

You can exclude categories from a dimension directly from the **Dimensions** pane. You do not have to create a dimension view.

The **Exclude** option not only omits categories and their descendants from the dimension view; it also omits the data associated with those categories.

For example, suppose you want to exclude Northern Europe, Southern Europe, Asia Pacific, United States, Brazil, and Mexico categories and their descendants from your cube, so that users only see data for the United Kingdom and Canada.



You create the dimension view and apply it to the **PowerCube** object. All data associated with the excluded categories is omitted from the resulting cube and its related reports.

Excluded categories are not included in category counts, and values associated with them are not included in rolled-up measure values. This differs from the **Cloak**, **Summarize**, and **Suppress** options, which act on the categories but have no effect on the data written to the cube.

In dimensions with alternate drill-down paths, excluding categories from one path excludes the same categories from all other drill-down paths.

To support the calculation of relative time categories for the cube, do not suppress or exclude weekend days (Saturdays and Sundays) and their associated measure values.

Steps

1. Open the **Categories** diagram for the dimension that contains the categories or level to be excluded and, if you have not already done so, add a dimension view.

   For more information, see "Create a Dimension View" (p. 159).

2. In the **Dimensions** pane, select your new view.

3. In the category viewer, select the categories or level to be excluded and, from the **Diagram** menu, click **Exclude**.

4. Open the **PowerCube** property sheet, and click the **Dimensions** tab.

5. In the **Dimensions** list, right-click the dimension in which you created the dimension view and click **View**.

6. In the **Select a View** dialog box, select the dimension view that you created, and click **OK** twice.

Omit Descendant Categories Using Summarize

When you create a dimension view and use the **Summarize** option on a category, you remove the data associated with the descendants of the selected category, but roll up their measure values to the summarized level. When you summarize all categories in a level, if new subordinate categories are added, they are automatically included in the summarized ancestors.

Although applying a summarized view consolidates the records in a cube by default, you should not use this option as a means of decreasing the size of your cube. Also, you cannot apply summarized views to a time dimension when the model contains a measure with **Time state rollup** applied.

Steps

1. Open the **Categories** diagram for the dimension that contains the categories or level to be summarized and, if you have not already done so, add a dimension view.

   For more information, see "Create a Dimension View" (p. 159).

2. In the **Dimension** pane, select your new view.

3. In the category viewer, select the categories or level to be summarized and, from the **Diagram** menu, click **Summarize**.

   This action will omit the descendant categories but retain their combined rollup value.

4. Open the **PowerCube** property sheet and click the **Dimensions** tab.

5. In the **Dimensions** list, right-click the dimension in which you created the dimension view and click **View**.

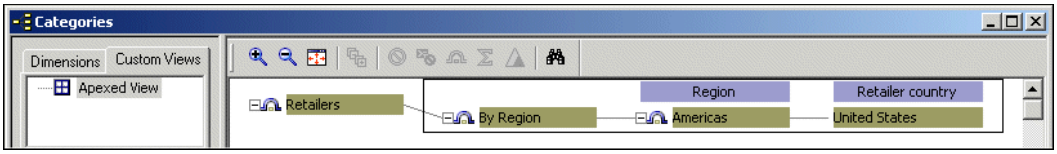
6. In the **Select a View** dialog box, select the dimension view that you created, and click **OK** twice.

Omit Categories Using Apex

When you create a dimension view and use the **Apex** option on a selected category, the resulting cube contains only the apexed category and its immediate descendants. The ancestors of the category, any siblings, and any descendants of these siblings are all omitted.

Any special categories that include descendants of the apexed category are connected to the apexed category as if it were the root category.

For example, suppose you apply the **Apex** option to the United States category in the Region dimension. The resulting PowerCube and its associated reports only show data for the United States and its descendants.



You cannot perform an apex operation at the union with an alternate drill path.

Steps

1. Open the **Categories** diagram for the dimension that contains the categories to be apexed and, if you have not already done so, add a dimension view.

   For more information, see "Create a Dimension View" (p. 159).

2. In the **Dimension** pane, select your new view.

3. In the category viewer, select the categories to be apexed and, from the **Diagram** menu, click **Apex**.

   Note: This action omits all ancestor and sibling categories. If you need to reverse this command, reapply it at a higher level.

4. Open the **PowerCube** property sheet, and click the **Dimensions** tab.

5. In the **Dimensions** list, right-click the dimension in which you created the dimension view and click **View**.

6. In the **Select a View** dialog box, select the dimension view that you created, and click **OK** twice.

Setting Up Drill-through Targets

Drill through provides a mechanism for providing your report and analysis users with further details or more timely perspectives on the data without burdening the system resources with intensive data source queries.

Drill Through in IBM Cognos 8

In IBM Cognos 8, drill-through targets are created in IBM Cognos Connection using package-based (or model-based) drill-through definitions.

There are several benefits to authoring and storing the drill-through paths in IBM Cognos Connection instead of in the Transformer model:

- The Transformer modeler does not need to specify any target reports or settings in the model for the purposes of drill through.

- When changes to drill-through paths are required, the PowerCube does not have to be rebuilt or updated.

  As a result, end users are less affected by cube rebuilds or updates.

- Drill-through paths can be defined by report or analysis users.

  Because report or analysis users can define drill-through paths according to their needs, they are less dependant on the Transformer modeler.

For information about specifying drill-through targets in IBM Cognos 8, see the IBM Cognos 8 *Administration and Security Guide*.

Considerations for Designing Drill-through Targets in IBM Cognos 8

When you design a PowerCube for use in IBM Cognos 8, you should understand concepts such as conformed dimensions and Member Unique Names (MUNs). This understanding will help you create the best environment for successful drill-through applications when your report and analysis users design drill-through paths for their own business needs.

For information about designing data for drill through in IBM Cognos 8, see "Designing Successful IBM Cognos 8 PowerCubes" (p. 44).

Note: In Transformer version 7.x, when the PowerCube was to be used in an IBM Cognos 8 drill-through definition, the Transformer modeler was required to enable the **Allow drill through for this PowerCube** check box on the **Drill Through** tab on the **PowerCube** property sheet. This ensured that the source value was passed as the business key during PowerCube to relational scenarios. In Transformer version 8.x, this is not required; the source value will always be passed as the business key to relational target reports.

Drill Through in IBM Cognos Series 7

Drill-through settings in Transformer version 8.x models are maintained for building PowerCubes for use in IBM Cognos 8 Business Intelligence Mobile Analysis or in IBM Cognos Series 7 products. This is possible only when the cube is designed without custom views.

Note: Setting the drill-through properties in a Transformer version 8.x model has no effect on drill through in IBM® Cognos® 8 Business Intelligence Reporting or IBM® Cognos® 8 Business Intelligence Analysis.

Drill-through Targets

The following drill-through actions are supported for PowerPlay and IBM Cognos 8 Business Intelligence Mobile Analysis:

- between a PowerCube (.mdc file) and an Impromptu Query Definition (.iqd) file
- from one cube to another, to overcome row or category limitations or to link cubes that cannot be created from one model because they are stored in different places or updated at different times
- between a PowerPlay report (.ppr or .ppx file) and an Impromptu report (.imr file), to show users the same or different measures (such as count) to the transaction level of detail, when only summary data is included in the cube

Drill Through with Impromptu Query Definition Files

If your model uses an Impromptu Query Definition (.iqd) file as a data source, and IBM Cognos Impromptu is installed, Transformer automatically adds the corresponding Impromptu report (.imr file) to the drill-down list for each measure. The correct filter is passed to display the appropriate Impromptu data, according to the dimension line in the OLAP report.

Drill Through Without Impromptu Query Definition Files

If your model is not based on an .iqd file, or if you want to view details of destinations other than .imr reports, you can add the target files to one of the following drill-through lists:

- the list on the **Drill Through** tab of the **Measure** property sheet, if you are restricting the drill-down capability to a single measure
- the list on the **Drill Through** tab of the **PowerCube** property sheet, if the drill-down capability applies to any point in the cube

Drill Through Constraints

You can use the Transformer Windows interface or Model Definition Language (MDL) to set up drill through to a wide variety of target reports simply by adding the destination filenames to the appropriate drill-through list. However, if the target file is renamed to make it more meaningful to your users, you must update the drill-through information in your model, then recreate and redeploy the cube.

If a drill-through filter applies to a level that was not declared unique in the model, the parameters passed to the target component must contain information for that level and for each ancestor level until a unique level or the highest level in the dimension is reached.

In PowerPlay-to-Impromptu drill-through operations, this is automatic. For example, if a category called Burlington belongs to the level City, which was not set to **Unique** on the **Level** property sheet, the filter contains City=Burlington plus the next higher level, State=Massachusetts.

For time dimensions, measure values are specified only for the most detailed leaf categories. Categories in Transformer time dimensions are expressed as ranges from the first to the last descendant. For example, the year 2006 is represented as 20060101-20061231.

For special categories and categories added manually, all child categories are handled successively until the source level is reached.

Add or Remove Drill-through Targets

Because cubes and other drill-through files may be created in different places at different times, their data and related information, or metadata, may not always match the information in the original cube. Ensure that you are linking to the most up-to-date information.

When a model is based on an Impromptu Query Definition (.iqd) file, the corresponding Impromptu reports (.imr) automatically appear in the drill-through list for each measure in the source. You can remove some or all of these reports from the **Measure** property sheets, or add additional reports that are not associated with the original source.

To change targets for all measures in a cube at once, use the **Drill Through** tab on the **PowerCube** property sheet.

Example - Setting up a Combination of Drill-through Targets

To meet the needs of various users, you may need to set up multiple drill-through targets:

- Drill through from a higher-level cube, showing the results for an entire product line, to a cube that has the same measures and dimensions but has additional levels in the Product dimension.

  Users can drill through to the detailed cube to analyze results based on different attributes, such as color or model.
- Drill through to a cube having the same degree of detail, but differing dimensions.

The drill-through cube may be separated from the original cube in time and space. For example, your regional office maintains client addresses in a supported target report. Add this report to the drill-through list on the **PowerCube** property sheet so that your users can see address details in their OLAP report when they select any cell.

- Drill through to a cube having the same dimensions, but not all measures.

  For example, a model based on an .iqd file has the measures Profit Margin, Revenue, and Cost, and the drill-through report shows all three. You remove the .imr report from the property sheet of the Profit Margin measure.

- Drill through to only those reports that are in scope, excluding all others based on a set of restrictions established in the cube model.

Steps

1. Open the **Measure** property sheet, and click the **Drill Through** tab.

   If you are setting up drill through for all measures in a cube, open the **PowerCube** property sheet, and click the **Drill Through** tab.

2. If you are setting up drill through for the first time, click the **Allow drill through for this measure** check box.

   For PowerCubes, the equivalent setting is **Allow drill through for this PowerCube.**

3. To add a target, click **Add**, browse to the target file, and click **Open** to add the file to the custom drill-through list.

   Tip: You can then select the target and click **Modify** to make any required changes. Or, to delete an item from the target list, select it and then click **Remove**.

4. If desired, type explanatory text in the **Report description** box.

5. Click **OK**.

Restrict Drill-through Targets by Level

To narrow the scope, you can restrict drill through from a measure or from a cube, by level.

Steps

1. In the **Dimension Map**, from the **Edit** menu, click **Show Scope** and then select **DrillThrough**.

2. In the first drop-down list box, click the measure or cube on which to restrict drill through by level.

3. In the second drop-down list box, click the drill target for which the restriction will apply.

4. In the **Dimension Map**, click the level or dimension for which you want to restrict drill through.

5. Right-click the dimension and click **Exclude DrillThrough**.

   The interface highlighting shows which level or dimension is now unavailable, or out-of-scope, for drill through. By default, the unavailable item appears on a red background, which is the formatting used for a source with missing columns.

6. Build the cube by clicking **Create PowerCubes** from the **Run** menu, and open it in your OLAP reporting component. Confirm that the level or levels you excluded do not support drill through.

   Note: If your users attempt to drill through while positioned in an excluded level, they will see a warning message similar to the following:

   There are no drill-through targets available. Verify your selection or confirm that drill through is available at this level.

Add Drill-through Targets to UNIX/Linux Cubes

You can set up drill through to a cube on a UNIX/Linux server by creating a temporary file in the **Custom reports** list and entering the path of the targeted drill-through file.

Steps

1. Open the **PowerCube** property sheet and click the **Drill Through** tab.
2. Click **Add**, type the name of your temporary file in the **File name** box, and click **Open** to add it to the **Custom reports** list.
3. Select your temporary file from the **Custom Reports** list, and click it once to edit the filename.
4. Type the path to the UNIX/Linux server, and press the Enter key on your keyboard.

Use Alternate Data Sources for Cube Creation

One way to customize cubes for individual audiences is to change the source data. You can create two different cubes based on the same model, but alter the content by using different source files.

The structure of the alternate source file must exactly match the structure of the source file on which the original model is based. The alternate file must have the same columns as the original for it to be read into the model. Also, the alternate data source must contain measure values. Purely structural alternate data sources will be ignored during a cube build.

For example, suppose you decide to build cubes for different sales divisions from different source files. In all cases, you can use the same sales analysis model. You use your database access component (such as Framework Manager or IBM Cognos Impromptu) to set up several alternate source files. Each file contains measure values, each is structurally identical to the source file on which the model is based, and each filters the data to retrieve values for only one sales division.

Steps

1. Open the property sheet for the PowerCube that will use the alternate source file and click the **General** tab.
2. In the **Data source list** box, select the appropriate item in the alternate source file.
3. In the **Source file** box, type or browse for the name of the alternate source file, and click **OK**.

Build a Subset of Your Cube Data for Testing Purposes

You can generate categories and create a cube that uses a subset of your source data, to obtain a smaller, more quickly built cube for testing purposes.

For example, suppose your model uses the following database sources:

- Customers (9,400 structural records)
- Products (1,720 structural records)
- Daily Sales (350,000 transactional records)

To create a quick test build, include only the first 350 records from each data source.

Note: When the number of test rows in the data source is limited to slightly more or fewer than 15 rows, Transformer generates extra categories in the model. The extra categories are labeled with numbers instead of proper category labels. This problem exists only when running test builds; when the complete cube is built, Transformer correctly processes all the rows from all the queries.

Steps

1. From the **Run** menu, click **Test Build**.
2. In the **Number of records to process** box, type the number of records to be read from the source.
3. Click either **Generate categories only** or **Create PowerCubes**.
4. Click **OK**.

Update the PowerCube Metadata

During PowerCube creation, all the source data is read, categories are generated in accordance with the model structure, and the data is written to a cube file (.mdc) for use in the OLAP reporting components.

There are several factors that affect the time it takes to create a PowerCube, including partitioning, cube optimization, incremental updates, the size of the cube, and, most importantly, the number of data sources.

There is other data in the model that is not dependent on the category generation process. Referred to as metadata, this related information includes

- object names
- short names
- descriptions
- drill-through items
- custom views

You can use the **Update Selected PowerCube** command from the **Run** menu to quickly update this related information for an existing cube without affecting the main cube data.

For example, suppose your PowerCube production schedule involves recreating a cube monthly, while your transactional data source is updated weekly. You set up drill through so your users can see, on the first week of each month, the first weekly report; on the second week, the first and second weekly reports; and so on. Although you only create the full PowerCube once a month, you can update its related information weekly by adding the extra drill-through report for the required measure.

Steps

1. In the **PowerCubes** list, select the cube whose metadata you want to update.
2. From the **Run** menu, click **Update Selected PowerCube**.
3. In the **Update PowerCube** dialog box, select the items you want to update.
4. Click **OK**.

Check Cube Status

After your PowerCubes or cube groups are created, you can check their status before you deploy them to your OLAP report users. You can also apply a filter in the **PowerCube Status** dialog box to focus on the cubes that need attention at any particular point in the cube customization process.

Steps

1. Click the **PowerCubes** list to make it active and then, from the **Tools** menu, click **PowerCube Status**.
2. In the **PowerCube Status** dialog box, select the cube whose staus you want to view.

   The status value is listed in the **PowerCube Properties** box.

Chapter 7: Adding Security

Transformer supports simultaneous user authentication and logon using the full range of supported IBM Cognos 8 security providers. You can also add custom views to each PowerCube to grant or deny access to sensitive business intelligence information. These access controls can be customized down to the query object level: not merely to reports and cubes, but to the specific levels, categories or members, and measures within them.

Choosing the Type of Security to be Applied

Transformer supports two types of security to restrict data access across the IBM Cognos 8 reporting components: member-based security and cube-based password protection.

When you use member-based security, you create custom views and apply these views to specific categories (members), dimensions, or components thereof. This filters the cube data that is shown to specific report users. Member-based security uses security objects such as users, groups, or roles, to define user access to information.

With cube-based security, you apply security to an entire PowerCube or cube group by setting a password to restrict access to authorized users.

Assessing Your Security Requirements

Use the following questions to assess the need to control access to information and to identify the specific security levels to apply to each user, group, or role.

- Can you place your users into distinct groups based on their information needs and access privileges? Or, do people change jobs or locations so frequently that this is not practical?
- If your organization already has user groups or roles, are these based on network and database access, or existing Human Resources classifications such as job functions and task profiles? Will you need to realign these user groups to more accurately reflect decision-making roles?
- Can you rely on database or network operating system logins to restrict access, or must you implement alternative security for your sensitive data? Do you have directory-based security already in place?

After you decide on the necessary levels of security to use, the process of adding security to your models and cubes consists of the following tasks:

- ensuring that the required authentication provider is configured in your IBM Cognos 8 Business Intelligence environment and that the required users, groups, and roles are available from that IBM Cognos 8 namespace, referencing the configured authentication provider of your choice

  For more information, see the IBM Cognos 8 *Administration and Security Guide*.
- assigning the security objects from the security namespace configured in IBM Cognos 8 to custom views, and then combining custom views with dimension filtering to appropriately subdivide your business information

- associating access controls with your PowerCubes before delivering them to your users

To begin, consider the business reasons for restricting access to data. For example, you may have confidential data that only specific users are allowed to see. Or your configured data source may contain a large amount of information, and your users need to retrieve data from only specific dimensions or levels. Perhaps you have a dimension that contains many categories or members, and your users need to retrieve only a subset of records from that dimension.

Depending on your data source, the underlying database security may also affect user access to certain categories of information. Therefore, assigning access to a level may not guarantee that the user also has access to all the categories or members in that level.

Before you add security in Transformer, ensure that security was set up correctly in IBM Cognos 8. For more information, see the IBM Cognos 8 *Administration and Security Guide*.

IBM Cognos 8 Security Objects

Users, groups, and roles are IBM Cognos 8 security objects created for authentication and authorization purposes. You can add groups created in authentication providers, or you can create your own in IBM Cognos 8.

Users

A user entry is created and maintained in an authentication provider to uniquely identify a human or a computer account. You cannot create users in IBM Cognos 8.

Information about users, such as first and last names, IDs, passwords, locales, and e-mail addresses, is stored in authentication providers.

Users can become members of groups defined in authentication providers and groups defined in IBM Cognos 8. A user can belong to one or more groups. When users are members of more than one group, their access permissions are merged; this is known as the union of views principle.

Groups and Roles

Groups can include individual users, as well as other groups. Group membership is part of a user's basic identity. Users log on with all the permissions associated with the groups to which they belong. Examples of groups are Employees, Managers, and Sales Personnel.

A role is a grouping that typically includes users who have similar responsibilities and privileges in your organization. Roles can include users, groups, and other roles.

Individual users can belong to several groups or roles.

Group and role names must be unique.

Create Member-based Security

You add member-based security to a model by creating custom views of the data, and then assigning the custom views to individual cubes. When you create a custom view, you select security objects (users, groups, and roles) configured in your IBM Cognos 8 namespaces, and then define a specific view of the data for those security objects using dimension filtering methods, such as apexing or cloaking.

The security you specify for an object is passed on to the drill-though reports that reference that secured object. Only users with permission to see the secured object are able to see it in the published cube.

With Transformer, you can also create hierarchical custom views. Hierarchical custom views allow you to restrict the data accessible in the descendant views, without having to recreate full custom views. This is similar to how user class views were used in Transformer version 7.x.

You can add security to a model and update the model security at any time.

When you make changes to the security objects in your configured IBM Cognos 8 namespaces or source authentication providers, you do not need to rebuild the cube to reflect the changes. PowerCubes reflect the applied member-based security at run time. For example, if you use a group called System Administrators in a custom view within Transformer, and then change the users who belong to that System Administrators group in the original authentication provider, the PowerCube reflects the changes without needing to be rebuilt.

When you add or make changes to custom views, you must rebuild the cube for the changes to take effect.

Consider the following:

- You can reuse custom views on any cube within a model.
- If your model makes use of cube groups, member-based security is not inherited from the control cube. You must define member-based security for each member cube individually.
- With hierarchical custom views, each descendant inherits the parent view.
- Individual groups or roles cannot appear more than once in the same hierarchical custom view.
- Each custom view must be applied individually to the cube. The only exception is when you apply a descendant custom view to a cube. In this situation, the parent views are automatically added to the cube.

Steps

1. Open a model in Transformer and, from the **Security** menu, click **Log On**.
2. In the **Logon** dialog box, select the appropriate namespace from the drop-down list, and log on as an authenticated user.
3. Open the **Categories** diagram for your model and click the **Custom Views** tab.
4. Right-click the **Custom Views** pane and click **Create Custom View**.
5. In the **Custom view name** box, type a name for the custom view.
6. If you are not logged on to the namespace from which you want to assign members, click **Log On**, select the namespace from the drop-down list, and then log on as an authenticated user for that namespace.
7. In the **Custom View** dialog box, click **Assign Security**, and then do the following:

- In the **Select Users and Groups** dialog box, click the appropriate namespace to expand the available group and role entries.

  Tip: To select individual users in the list, click the **Show users in the list** check box.

- From the **Available entries** list, select the security objects (roles, groups, and users) to be assigned membership in the custom view and click the right-arrow button.
- When the entries you want appear in the **Selected entries** box, click **OK** twice.

The new custom view appears in the custom view list, the model dimensions appear in the **Dimensions** list, and the measures appear in the **Measures** list.

8. To omit the dimension from the custom view, select the dimension and then click the **Omit** button .

   Tip: To reset a dimension to the parent or default view, select the dimension and then click the **Reset** button .

9. To exclude a measure from the custom view, in the **Measures** list, clear the check box for that measure.
10. In the **Custom Views** list, select the new custom view.
11. In the **Dimensions** list, select a dimension.
12. To create a custom view using the view operations **Exclude**, **Cloak**, **Suppress**, **Apex**, and **Summarize**:

    - Click the **Customize** button .
    - In the category viewer, right-click the categories and apply the view operations (**Exclude**, **Cloak**, **Suppress**, **Apex**, and **Summarize**) until you achieve the desired effect.

      For information about Transformer view operations, see "Customizing Cube Content with Views" (p. 158).

    Tip: To reset a dimension to the parent or default view, select the dimension and then click the **Reset** button .

13. When you are finished applying view operations to the custom view, close the **Categories** diagram and then, from the **Custom Views** list, drag the custom view that you want to associate to the cube to the appropriate cube or cube group in the **PowerCubes** list.

Update Model Security

When the security data in your authentication provider is updated, you are required to update the security objects associated with Transformer custom views.

When custom views are enabled for a model, Transformer reads the security objects from the namespace configured in IBM Cognos 8. Because there is no automatic synchronization of security objects between Transformer custom views and your configured IBM Cognos 8 namespace, when the hierarchy of the associated security data is changed, or users are deleted, the model gets updated but the associated custom views become invalid.

To update the security objects in your model, confirm that the users, groups and roles are valid in the IBM Cognos 8 configured namespaces, and then adjust your custom views accordingly.

Note: When you remove users, groups, or roles from your model, they are not deleted from your authentication provider or configured IBM Cognos 8 namespaces.

Step to Remove Obsolete Security Objects

- From the **Security** menu, click **Remove Obsolete Security Objects**.

The security objects are updated, and groups, roles, and users no longer present in the configured IBM Cognos 8 namespaces are removed from the custom view.

Steps to Remove Security Objects From a Model

1. From the **Security** menu, click **Show Security Objects**.

   The **Security Object Management** dialog box shows the security objects imported into the model, and their associated custom views.

2. In the **Security objects imported into model** list, select the security object you want to remove from the model and click **Unassign Custom View**.

3. When you finish removing the security objects from the custom view, click **OK**.

Union of Custom Views

For the sole purpose of migration from IBM Cognos Series 7 security to an alternate security provider, Transformer allows you to place security objects from different namespaces within a single custom view. IBM Cognos 8 does not support secured PowerCubes that have been associated with multiple namespaces except during the testing of migration. We do not support the deployment of cubes secured against multiple namespaces within a production environment because IBM Cognos 8 does not indicate to end users when they are not accessing the full view of the cube. For this reason, when you deploy cubes for use in production, ensure that they are secured against only one namespace by selecting the **Restrict PowerCube authentication to a single namespace** option in the **Signon** properties in the **Publish PowerCube as Data Source and Package** wizard.

A user can be a member of several user groups or roles, and can belong to multiple custom views. In IBM Cognos 8, when a user belongs to more than one group or role and the group, roles, or user belongs to multiple custom views, the cube opens with a union of all the custom views.

A union of custom views combines the access capabilities of its member custom views, but not their restrictions. The rules for determining what is shown in the cube as a result of a union of custom views are

- that which is shown to an individual custom view is shown to the union

  That which is not shown to an individual custom view is the same as that which is not shown to the union.

- a category is shown to the union if it is shown in at least one custom view

- measure values, or cell values, are shown to the union if there is at least one custom view that sees all the categories in the cell domain

  A measure value, or cell value, is not shown to a union when none of the custom views in the union can see all the categories in the cell domain.

- in a union of custom views, the root category shifts to the lowest common ancestor of all member custom views

These rules apply to all custom views in the union. The following examples illustrate the union of custom views in different scenarios.

Example - Two Custom Views, Each with an Apex in the Same Dimension

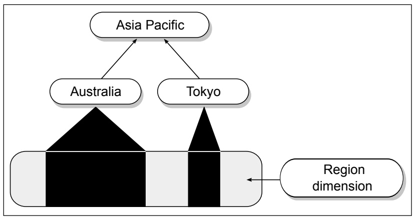
This example describes the union of two custom views that each have an apex in the same Region dimension.

Custom view A: apex Australia

Custom view B: apex Tokyo

There is no visible common root for the two apexed categories. When the cube is opened with the union, the lowest common ancestor of the apexes is shown: Asia Pacific. In accordance with the roll-up rule, the rolled-up value at Asia Pacific will be Australia and Tokyo.

Note: If the PowerCube has the **Block totals for parents with excluded children** option set on the **Processing** tab of its property sheet, the value of Asia Pacific is N/A.



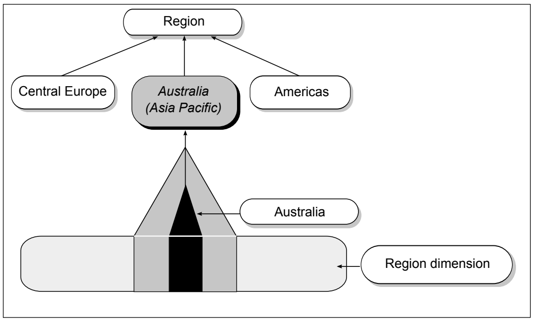
Example - Apexed Custom View Cloaked by Another Custom View in the Same Dimension

This example describes the union of an apexed custom view cloaked by another custom view in the same dimension.

Custom view A: apex Australia

Custom view B: cloak Asia Pacific

The Region dimension is temporarily modified. Australia becomes a peer of Central Europe and Americas, in the place of Asia Pacific, which is not shown. All values are shown, although the other Asia Pacific categories are shown indirectly because the totals for those categories roll up to the Region dimension total.



The values for Korea, China, Japan, and Taiwan are cloaked. As a result, the values for those categories are visible only in the total for the Location dimension.

Example - Apexed Custom View Excluded by Another Custom View in the Same Dimension

This example describes the union of an apexed custom view excluded by another custom view in the same dimension.

Custom view A: apex Australia

Custom view B: exclude Far East

The Region dimension is temporarily modified. Australia becomes a peer of Central Europe and Americas, in the place of Asia Pacific, which is not shown. The values for the other Far East categories are not shown because they are excluded from the Region dimension total.

This example is similar to the previous example; however, in this case, the values for Korea, China, Japan, and Taiwan are excluded from the total for the Location dimension.

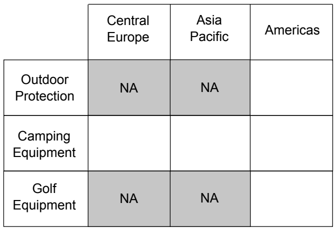
Example - Union of Two Apexed Custom Views in Different Dimensions

This example describes the union of two apexed custom views in different dimensions.

Custom view A: apex Camping Equipment

Custom view B: apex Americas

All children of the Region dimension are shown because they are shown to custom view A. The sibling categories of Camping Equipment are shown because they are shown to custom view B. Data values visible to either of the two custom views are shown. In a crosstab report, the cells for categories outside of the apex, such as Golf Equipment, Central Europe, are shown but the data values appear as **NA**.



	Central Europe	Asia Pacific	Americas
Outdoor Protection	NA	NA	
Camping Equipment			
Golf Equipment	NA	NA	

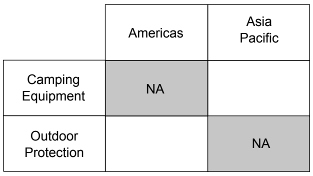
Example - Union of Two Multi-dimensional Apexed Custom Views

This example describes the union of two apexed parent dimensions with multiple categories.

Custom view A: apex Camping Equipment, apex Asia Pacific

Custom view B: apex Outdoor Protection, apex Americas

A temporary common ancestor of each apex is shown in each dimension: Region and Product. The Camping Equipment for Asia Pacific is shown, as is Outdoor Protection for Americas. All other categories show **NA**.



	Americas	Asia Pacific
Camping Equipment	NA	
Outdoor Protection		NA

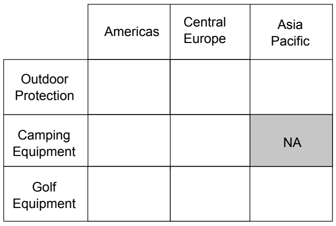
Example - Union of Excluded Categories in Two Dimensions

This example describes the union of excluded categories in two dimensions.

Custom view A: exclude Camping Equipment

Custom view B: exclude Asia Pacific

Because custom view A is shown all categories except the Camping Equipment, and custom view B is shown all categories except Asia Pacific, the union shows all categories. Each custom view is shown the cells of the siblings of the excluded categories. The only area not shown is the intersection of the two exclusions: Camping Equipment and Asia Pacific.



	Americas	Central Europe	Asia Pacific
Outdoor Protection			
Camping Equipment			NA
Golf Equipment			

Examples - Unions of Excluded Categories in the Same Dimension

The following examples describes the union of excluded categories in the same dimension.

Example 1

Custom view A: exclude Central Europe

Custom view B: exclude Americas

Each custom view is shown the values excluded by the other custom view.

Example 2

Custom view A: exclude France

Custom view B: exclude Central Europe, Mexico

France is the only category that is not shown in either custom view. The other children of Central Europe, and Mexico are shown to custom view A.

Example 3

Custom view A: exclude France, exclude Americas

Custom view B: exclude Central Europe, exclude Mexico

France and Mexico are not shown in either custom view. The other children of Americas are shown in custom view B. The other children of Central Europe are shown to custom view A.

Example - Union of Excluded and Cloaked Categories in the Same Dimension

This example describes a dimension with custom views that include excluded and cloaked categories.

Custom view A: exclude United States, cloak Mexico

Custom view B: cloak Boston

In each custom view, Boston is cloaked. Because custom view B is shown all Region values and all categories except Boston, the union of the two custom views eliminates the restrictions on all categories except for Boston for custom view A.

Example - Union of a Custom Views with Omitted Dimensions

This example describes the union of custom views with omitted dimensions.

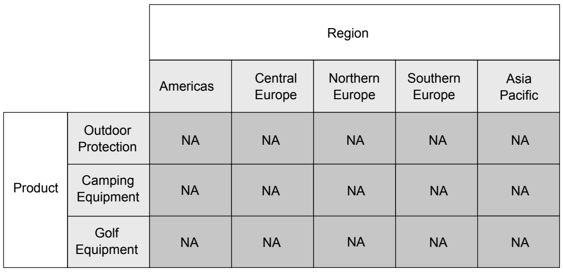
Custom view A: omit Region dimension

Custom view B: omit Product dimension

Custom view A is not shown any category in the Region dimension, and this dimension is omitted from the domain of any query formed by custom view A. The PowerCube query engine replaces the gap with the root of the Region dimension. Custom view A is shown all other dimensions, including the complete Product dimension.

Custom view B is not shown any category in the Product dimension, and this dimension is omitted from the domain of any query formed by custom view B. The PowerCube query engine replaces the gap with the root of the Product dimension. Custom view B is shown all other dimensions, including the complete Region categories.

The union of these two custom views results in all categories of all dimensions being shown. However, because a cell value is visible to the union only when there is at least one custom view that can see all the categories in the cell domain, only values of cells that are of the root of Region or Product are shown. Drill down produces **NA** values.



		Region				
		Americas	Central Europe	Northern Europe	Southern Europe	Asia Pacific
Product	Outdoor Protection	NA	NA	NA	NA	NA
	Camping Equipment	NA	NA	NA	NA	NA
	Golf Equipment	NA	NA	NA	NA	NA

Create Password-protected Cubes

You apply security to an entire PowerCube or cube group by setting a password to restrict access to authorized users. When users access cubes and reports in their reporting component, they must enter the password in the **Cube Logon** dialog box to see the data.

Access is controlled using database signons and, provided your users have configured their authentication source for auto-access of protected cubes, they are not repeatedly asked to identify themselves during the same run-time session.

When you publish a data source to IBM Cognos Connection and include the cube password association in the data source, users are not prompted for the cube password when they log on. For information about importing data sources in IBM Cognos Connection, see the IBM Cognos 8 *Administration and Security Guide*.

When a password is defined at the root node of a cube group, the same password applies to all cubes in the group. However, a password defined for a member of a cube group overrides the password defined at the root level for the group.

Password-protected PowerCubes are recommended if you intend to build cubes for disconnected or mobile use. IBM Cognos 8 secured PowerCubes are not available for use in IBM Cognos 8 Business Intelligence Mobile Analysis or IBM Cognos Series 7 versions of PowerPlay for Windows.

Steps

1. In the property sheet for the PowerCube or cube group, click the **Output** tab.

2. In the **Password** box, enter the password and click **OK**.

3. In the **Confirm Password** dialog box, re-enter the password and click **OK**.

Combining Custom Views with Dimension Views

You can add custom views to a cube that already has dimension views. Use this combination when you must control access to certain information (such as salaries) based on a user's security classification, but do not need to restrict access to the remaining data.

For example, suppose you create a cube group that includes separate cubes for Central Europe, the Americas, and the Far East. For each cube, additional protection is imposed by creating custom views that group OLAP report users separately from security administrators for each region. When the cubes and the accompanying authentication sources are released to users in each region, regional security administrators can maintain security definitions for their own regions.

We do not recommend using dimension views as the basis for custom views if auto-partitioning is in place. In very large cubes with complex partitioning schemes, test your design to ensure that combination views do not increase cube creation time.

Combine custom views with dimension views by performing the following tasks:

❑ Create all the cube groups and dimension views you need for the model.

❑ Create and apply custom views to the objects in the **PowerCubes** list.

❑ On the **Dimensions** tab of the **PowerCube** property sheet, select the appropriate dimension views.

❑ Create and deploy your secured PowerCubes.

Block Total Values for Parent Categories with Excluded Children

You can block the summed values for parent categories with excluded children, so that the resulting reports show a Denied value rather than the totals of the non-excluded children. This prevents the display of data that is an inaccurate rollup of only non-excluded categories.

Missing values take precedence over denied values. Categories show zero, N/A (not available), a blank (nothing in the cell), or missing values, depending on how you specified that missing measure values be handled. This allows you to distinguish between missing and denied values.

For example, suppose you are responsible for a North American sales cube that has detailed data for Mexico, Canada, and the USA. A group of users are allowed to see the values for Mexico and the USA, but not the values for Canada, so you exclude the Canada category.

You also restrict the view for aggregate values, so that the users see Denied instead of the totals for North America. This prevents these users from deducing the values for Canada. You should be aware that selecting this option means that, should your users apply filters for North America with the contribution for Canada excluded, they may erroneously conclude that Canada had no sales.

Steps

1. In the **PowerCubes** list, right-click the cube you want to modify and then click **Properties**.

2. On the **Processing** tab of the **PowerCube** property sheet, select the **Block totals for parents with excluded children** check box and then click **OK**.

Upgrade an IBM Cognos Series 7 Secured PowerCube

You can open IBM Cognos Series 7 models with secured cubes in Transformer version 8.x, and upgrade the IBM Cognos Series 7 user class views and user classes for use in IBM Cognos 8.

If you want to move to an IBM Cognos 8 supported authentication provider other than Access Manager, you can do this over time.

When you open the IBM Cognos Series 7 secured model in Transformer version 8.x, you can choose to:

- Import the IBM Cognos Series 7 user class views associated with the model, but not the user classes.

  Choose this option when you want to maintain the view operations applied in the IBM Cognos Series 7 user class views but not use an IBM Cognos Series 7 namespace with the custom views, or if you do not intend to expose IBM Cognos Series 7 as an available namespace configured in IBM Cognos 8.

  Note: Prior to building and using the Transformer version 8.x cube in any of the IBM Cognos 8 Web studios, you will need to associate new security objects to the upgraded custom views.

- Import the IBM Cognos Series 7 user class views and user classes associated with the model.

  Choose this option when you want to maintain the view operations applied in the user class views and use the IBM Cognos Series 7 user classes, or if you want to transition to an alternate security provider but need to maintain the IBM Cognos Series 7 user class objects to ensure the transition is carried out correctly.

  This option requires you to configure the IBM Cognos Series 7 security on which the upgraded model was designed as an available namespace in IBM Cognos 8. The unique identifier that locates the user class in Access Manager is converted to an IBM Cognos 8 identifier, and this process will not be successful if you use this option with a different IBM Cognos Series 7 namespace.

- discard all existing custom views and security objects

  Choose this option when you plan to create new custom views and use only the security objects currently configured in the IBM Cognos 8 namespace.

Steps

1. From the Transformer Welcome page, click **Open a model**, browse to the location of the IBM Cognos Series 7 secured model, select the model and click **Open**.

   Tip: If you are already in Transformer, click **Open** from the **File** menu.

2. In the **Import model with IBM Cognos Series 7 user class view** dialog box, select the appropriate security import option and click **Next**.

3. If you selected **Import user class views and user classes from the model**, in the **Logon** dialog box, select the appropriate namespace and then log on with your user ID and password.

4. In the **Available namespace**(s) box, select the namespace used to secure the IBM Cognos Series 7 cube.

   Tip: If the namespace does not appear in the list, click **Logon As** to select and log on to the namespace.

5. Click **Finish**.

For PowerCubes that are in development and transitioning from an IBM Cognos Series 7 namespace to an alternate security provider, you can associate all the applicable namespaces on the **Data Source** tab of the **PowerCube** property sheet. This option is intended only for the testing of migration, and requires that the modeler or administrator log on to all the applicable namespaces prior to accessing the PowerCube package in IBM Cognos 8. Failing to log on to all applicable namespaces will result in an inaccurate view of the data. This feature is not supported for the deployment of cubes for end users.

For information about PowerCube data source properties, see "Publish a PowerCube" (p. 186).

Chapter 8: PowerCubes in Production Environments

With Transformer, you can build and deploy IBM Cognos PowerCubes in distributed production environments. This enables you to take advantage of dedicated Transformer servers for optimal cube build performance, while ensuring that your IBM Cognos 8 servers are fully available to your IBM Cognos 8 consumers.

Building and deploying PowerCubes in production environments involves

- managing languages and locales
- publishing cubes
- load balancing the IBM Cognos 8 servers
- maintaining models and cubes
- updating published cubes and cube connections

Managing Languages and Locales

The operating system locale setting determines the run locale for IBM Cognos 8 server requests, as well as the current code page for Transformer. To change the system locale on Windows, you must update the Regional and Language Options, including the language for non-Unicode programs. Changing these settings on Windows requires that you restart the computer. On UNIX systems, you can change the locale without restarting the computer.

To display the run locale Transformer is currently using, view the **Session Information** available from the **File** menu. For example, if the locale is set to **English (Canada)**, the run locale will be **en-ca**. Transformer sends the run locale in its requests to the IBM Cognos 8 server. When Transformer is requesting data from a multilingual package or report, the run locale may be used to determine the language in which data is returned.

Note: When a report is authored, the author chooses a content locale. This content locale is saved with the report and is the locale in which the metadata is expressed when Transformer accesses the report. As a result, ensure that reports used as data sources in Transformer have the appropriate content locale.

By default, Transformer builds PowerCubes in the current code page as determined by the locale settings. On many UNIX and Linux systems, the default code page is UTF-8. Transformer 8.x can build PowerCubes in the UTF-8 code page, which is a Unicode encoding. The resulting cube is not a multilingual cube; it is a unilingual cube in the UTF-8 code page. IBM Cognos Series 7 products are unable to read PowerCubes that use the UTF-8 code page. To build PowerCubes in a code page that is different than the current code page, you can change the **Cube code page** setting on the **Model** properties sheet. For example, you can build a windows-1252 code page PowerCube on Linux so that IBM Cognos Series 7 products can use the cube.

Transformer supports Unicode data sources, including Framework Manager packages, IBM Cognos 8 reports, and exported .csv files from IBM Cognos 8 reports. Transformer converts the Unicode data to the code page of the current locale for processing and display, and will also convert the data to a cube code page if one is specified. You must use compatible code pages, otherwise character loss may occur. For example, you should not process Japanese text data when Transformer is running in an English locale.

Publish a PowerCube

When your IBM Cognos 8 environment is properly configured, you can publish any object in the PowerCubes list from Transformer directly to IBM Cognos Connection. You must specify the Windows or UNIX/Linux location from where the IBM Cognos 8 server accesses each cube and, if user authentication is enabled, the external IBM Cognos-supported namespace to use.

You can set suppression options for the package published with the PowerCube. These options determine whether IBM Cognos 8 studio users can choose to hide empty rows only, empty columns only, or both empty rows and columns. Options also determine the types of empty values that can be suppressed, such as zero or missing values. Types of empty values that users can choose to suppress depend on the IBM Cognos 8 studio.

You can organize cubes efficiently by publishing in a way that hierarchies are represented in IBM Cognos Connection.

For time-based cubes, the parent and all child cubes are published, but a package is created for the parent cube only. This parent includes references to the child cubes.

Read Cache Size

The default read cache size for published PowerCubes is 80 MB. You can set this parameter to a value between 1 MB and 1 GB, as required for optimal query performance.

The optimal read cache size may be higher or lower than the default value of 80 MB. This is to be expected, as PowerCubes in production vary widely in type and query characteristics.

Note that the read cache size has no effect on the initial time required to open a cube.

The typical profile for query performance, or processing time, follows a pattern whereby performance increases with the read cache size and then levels off beyond the optimal setting. To determine the optimal setting, we recommend that you lower the default by 10 MB (or 5 MB, or 1 MB, depending on the level of fine tuning desired) and use the resulting query performance results as a guide for establishing whether further reductions, or increases, are required.

The optimal read cache size will change as the cube grows and changes in the production environment. As a result, you should review the optimal read cache size when changes to the user's query performance pattern, or changes in the PowerCube characteristics, occur.

You can publish Transformer version 8.x cubes in the following ways:

- from within Transformer

  You can publish a single cube using

  - the **Publish PowerCube** wizard if no other data source with the same name as this PowerCube's data source exists in the location you are publishing to

  - settings stored in the model
  - an .xml publish specification that you create from the **Run** menu and publish from the command line

    You can publish one cube, or all the cubes in the model.

- from within IBM Cognos Connection

  Use IBM Cognos Connection to deploy single cubes.

  For more information about deploying packages from within IBM Cognos Connection, see the IBM Cognos 8 *Administration and Security Guide*.

- from Framework Manager

  Use Framework Manager to publish multiple PowerCubes in a single package or mix your PowerCube data source with other data sources, such as a relational namespace. For more information, see the Framework Manager *User Guide*.

- from the command line

  For more information, see "Command Line Options" (p. 237).

You can disable cube publishing on the **Processing** tab of the **PowerCube** property sheet.

If you have already published a PowerCube and you want to replace it with a new version, you can copy and activate the cube (p. 210).

Steps to Publish a PowerCube Using the Wizard

1. In the **PowerCubes** list, right-click the PowerCube, and click **Publish PowerCube As Data Source and Package**.
2. In the **Publish cube** dialog box, select the **Start publish wizard** radio button and click **OK**.

   This option is available only if no other data source with the same name as this PowerCube’s data source exists in the location you are publishing to.
3. In the **Welcome** page of the IBM Cognos 8 **Publish PowerCube** wizard, click **Next**.
4. If prompted to log on, select the appropriate namespace, type your user ID and password and click **OK**.
5. In the **Name** box, specify a name for the published data source.

   This is the data source name that appears in IBM Cognos Connection.

   Tip: Optionally, you can add a description and screen tip for the published cube.
6. Click **Next**.
7. In the **Publish PowerCube** dialog box, do the following:
   - If you want to set a value that is different from the default of 80 MB, enter this value in the **Read cache size** box.
   - In the appropriate location box (Windows or UNIX/Linux), enter the location of the cube.

The location includes the full path and cube name.

This location must be the location where the IBM Cognos 8 server accesses and reads the cube. This may or may not be the same as the cube build location. If you build the cube locally, your administrator may have to set up a network share where you can ensure the cube is deployed, after it is built, for use in IBM Cognos 8 reporting and analysis. In this situation, you cannot test the data source until it is in the network share location.

- Under **Signon**, if the cube is secured, click the **Restrict PowerCube authentication to a single namespace** radio button, and select the appropriate namespace from the list.

  Secured PowerCubes published to production environments must use the **Restrict PowerCube authentication to a single namespace** option. When the cube includes security from more than one namespace, the **All applicable namespaces (including unsecured PowerCubes)** option is intended only for testing migration from an IBM Cognos Series 7 namespace to an alternate security provider. If you are testing the migration of IBM Cognos Series 7 PowerCubes and you want to review the security view that reflects the union of the applicable namespaces, you must log on to all applicable namespaces in IBM Cognos 8 prior to accessing the package. For more information, see "Upgrade an IBM Cognos Series 7 Secured PowerCube" (p. 183).

  If the cube does not include security, select the **All applicable namespaces (including unsecured PowerCubes)** option.

- If the cube is password-protected, select the **Cube password** check box and then, in the password boxes, type and confirm the cube password.

8. If you want to test the cube connection, click **Test the connection** and do the following:

   - If you are testing a secured cube, select the appropriate namespace, and enter the **User ID** and **Password**.

   - Click **Test**.

     If the cube connection fails, review the connection status message and take appropriate steps to resolve the connection.

     Note: If the cube contains custom views and you are logged on as Anonymous, you will receive an error message. This is because IBM Cognos 8 cannot validate the user "Anonymous" against the IBM Cognos 8 namespace user credentials. To avoid this situation, log on to the IBM Cognos 8 namespace with a user ID and password, and test the connection.

   - Click **Close**.

9. Click **Finish**.

10. In the **Finish** page, the **Create a package** check box is enabled and greyed out so that you can publish the package; click **OK**.

11. In the **Name** box, specify a name for the published package.

    This is the package name that appears in IBM Cognos Connection.

Tip: Optionally, you can add a description and screen tip for the published package.

12. If you want to change the location where the package is published, click **Select another location**, and browse to select the appropriate IBM Cognos 8 folder.

13. Click **Next**.

    If a package with the same name has already been published to IBM Cognos Connection, you are prompted to replace the existing entry or enter a new name for the published package. Click **OK** to continue.

14. If you want to change default suppression settings, select the appropriate options:

    - **Allow null suppression**. Select this option to make suppression options available to IBM Cognos 8 studio users when working with this package.

    - **Allow multi-edge suppression**. When null suppression is selected, this option determines whether users can suppress rows only, columns only, or both rows and columns. If this option is not selected, users can suppress rows only or columns only.

    - **Allow access to suppression options**. When null suppression is selected, this option determines whether users can choose the types of empty values that will be suppressed, such as zero or missing values. The types available to users depend on the IBM Cognos 8 studio.

    For packages that contain more than one cube, the selected suppression options apply to all cubes in the package. As well, these options apply to all users of a package, and cannot be set on an individual user basis.

15. Click **Finish**.

16. Click **OK**.

    Tip: You can enter the publish information in the **Data Source** and **Package** tabs of the **PowerCube** property sheet in advance, and use the wizard to publish the cube. The properties you set on these tabs will be populated when you use the wizard. If required, you can change the pre-set properties when you use the wizard.

Steps to Publish a Cube Using Settings Stored in the Model

1. In the **PowerCubes** list, right-click the cube, and click **Properties**.

2. Click the **Data Source** tab.

3. In the **Data source name** box, type the name of the cube.

4. In the **Windows location** box or the **Unix or Linux location** box, type the full path to the cube.

5. If the cube is secured, in the **Authentication** box, select the appropriate namespace.

   For PowerCubes that are in development and transitioning from an IBM Cognos Series 7 namespace to an alternate security provider, you can associate all the applicable namespaces. This option is intended only for the testing of migration, and requires that you log on to all the applicable namespaces prior to accessing the PowerCube package in IBM Cognos 8. Failing to log on to all applicable namespaces will result in an inaccurate view of the data. This feature is not supported for the deployment of cubes for end users.

Tip: If the cube is password protected, the **Create a signon** check box is selected.

6. In the **Description** box, type an optional description of the cube.

7. In the **Screen Tip** box, type an optional screen tip for the cube.

8. Click the **Package** tab.

9. In the **Name** box, type a name for the published package.

10. In the **Package** folder box, click the ellipsis (**...**) button and browse to the folder where you want the package to be published.

11. In the **Description** box, add an optional description of the package.

12. In the **Screen Tip** box, add an optional screen tip for the package.

13. If you want to change default suppression settings, select the appropriate options:

    - **Allow null suppression**. Select this option to make suppression options available to IBM Cognos 8 studio users when working with this package.

    - **Allow multi-edge suppression**. When null suppression is selected, this option determines whether users can suppress rows only, columns only, or both rows and columns. If this option is not selected, users can suppress rows only or columns only.

    - **Allow access to suppression options**. When null suppression is selected, this option determines whether users can choose the types of empty values that will be suppressed, such as zero or missing values. The types available to users depend on the IBM Cognos 8 studio.

    For packages that contain more than one cube, the selected suppression options apply to all cubes in the package. As well, these options apply to all users of a package, and cannot be set on an individual user basis.

14. Click **OK**.

15. In the **PowerCubes** list, right-click the PowerCube, and click **Publish PowerCube As Data Source and Package**.

16. In the **Publish cube** dialog box, click **Publish the cube using current settings**.

    The cube is published without using the wizard.

    Tip: If the data source or package for this cube already exists in IBM Cognos Connection, you are prompted to update the existing data source or package or create a new data source or package.

Steps to Create a Publish Specification to Publish One Cube

1. In the **PowerCubes** list, right-click the cube, and click **Publish PowerCube As Data Source and Package**.

2. In the **Publish cube** dialog box, click **Create publish specification** radio button, and click **OK**.

3. In the **File name** box, type the name for the publish specification file, then save the file.

4. Open the saved publish specification file in an XML editor, and enter the additional values for the publish parameters.

   Tips

   - If you defined publish parameters on the **Data Source** and **Package** tabs of the **PowerCube** property sheet, those values will appear in the publish specification file.

   - When you disable cube publishing on the **Processing** tab of the **PowerCube** property sheet, the cube will not be included in the publish specification file.

5. Save the file.

The information specified in the publish specification provides the necessary parameters for publishing the cube from the command line.

For information about publishing cubes from the command line using the publish specification, see "Command Line Options" (p. 237).

Steps to Create a Publish Specification to Publish All the Cubes in the Model

1. From the **Run** menu, click **Create a publish specification for all cubes**.

2. In the **File name** box, type the name for the publish specification file, and save the file.

3. Open the saved publish specification file in an XML editor, and enter the additional values for the publish parameters for all the cubes in the model.

   Tips

   - If you defined publish parameters on the **Data Source** and **Package** tabs of the **PowerCube** property sheet, those values will appear in the publish specification file.

   - When you disable cube publishing on the **Processing** tab of the **PowerCube** property sheet, the cube will not be included in the publish specification file.

4. Save the file.

The information specified in the publish specification provides the necessary parameters for publishing the cubes from the command line.

For information about publishing cubes from the command line using the publish specification, see "Command Line Options" (p. 237).

PowerCube Load Balancing

Load balancing of the IBM Cognos 8 servers is typically the responsibility of the administrator. As a modeler, you would not normally need to consider how the servers balance the load for query access. However, if you are building very large PowerCubes that will consume significant resources when accessed by consumers, or if you are building a PowerCube intended for users who must have high speed dedicated access, such as an organizational Chief Executive Officer or Vice President, you may want to confirm with your administrator that package routing is enabled.

After a PowerCube is built and is published to IBM Cognos Connection, your IBM Cognos 8 administrator can specify the individual servers in your IBM Cognos 8 environment to handle the PowerCube packages. Specifying the servers for Transformer cubes has the following advantages:

- For large cubes that require more server resources, performance can be optimized and other services in the server environment are not negatively impacted.
- Servers can be dedicated to specific users who require optimal cube performance.

  Each time these users access a cube, their requests are routed to the dedicated server.
- When you have several cubes that use the same metadata but are each built with a different language, you can use package routing to direct users who use a specific language to the appropriate PowerCube package.

  For information about package routing, see the IBM Cognos 8 *Administration and Security Guide*.

Maintaining Models and Cubes

You typically create several PowerCubes or cube groups from the same model. Each one may contain a different subset of the query objects defined in your model, aimed at different users.

These cubes are maintained for, and deployed to, various platforms, such as your UNIX/Linux environment, an IBM Cognos 8 Web studio, or a stand-alone Windows reporting component.

Your production Transformer models and cubes will probably require ongoing maintenance to respond to the evolving needs of your report users, changes in your data, or reorganizations in your workplace.

Whenever you make changes to your model and cubes, remember to do the following:

❑ Run **Check Model** to ensure that the updated data is consistent and error-free.

❑ Update security objects and related custom views as needed, before recreating your cubes.

❑ Update data source connection information for PowerCubes accessed by your OLAP report users.

If category code values in the PowerCube change, advise your cube consumers of the categories, or members, that changed so that they can better handle the impact on their reports. For more information about changes to category codes, see "Member Unique Names" (p. 45).

We strongly recommend that you keep a backup .mdl file format version of your Transformer model. If the .py? version or current .mdl version of the model develops conflicts or other issues, you will be able to revert to the backup. Models saved in py? file format are binary compatible models that can, over time, become corrupt. As a result, it is good practice to occasionally save your models as .mdl files, and save them back as .py? files again.

Source Data Updates

If you change the structure of your data by reordering or renaming columns or by switching to a new data source type, you may need to redesign a portion of your model before recreating your cubes. For more information, see "Match Model and Source Columns" (p. 195) and"Move Categories When Source Data Changes" (p. 196).

Model Updates

When your source contains records for time categories that are outside the range specified for the time dimension, Transformer creates the appropriate placeholder category: **Early Dates**, **Late Dates**, or **Invalid Dates**. To lock a dimension, preventing Transformer from adding unwanted categories, we recommend that, in the **Dimension** property sheet, you prohibit the automatic creation of new categories.

When you are making changes to your model, you should periodically clean up fragments that can cause generation errors by saving your model as an .mdl file rather than a .py? file. If your model has categories that are not used, we recommend that you delete them by running the **Clean House** command. For more information, see "Delete Inactive Categories" (p. 201).

The **Clean House** feature should be used with caution, as it may cause problems with consumers' reports in IBM Cognos 8. If you intend to delete inactive categories, we recommend that you advise your consumers so that they can remove those categories, or members, from their reports when they are directly referenced. Otherwise, they may receive an error the next time they run their reports.

Note: If your model contains incrementally updated cubes, you must retain all of their categories for the cubes to remain valid.

Cube Updates

You should alert your report users whenever you change a cube object, because the change may not be obvious when they look at the report.

When you enable the incremental update feature for a cube, Transformer appends the new data to the existing cube. When you disable incremental update, Transformer overwrites the existing cube. For more information, see "Set up Placeholder Categories for Cube Groups" (p. 209).

To rebuild an incrementally updated cube, you must delete the old .mdc file and recreate the entire cube.

For information about updating published cubes, see "Update Published PowerCubes and PowerCube Connections" (p. 210).

Automatic Creation of Multifile Cubes

Each PowerCube, or .mdc file, cannot currently be larger than 2 GB. If you plan to build a cube with data volumes that would result in an .mdc file larger than 2 GB, you can use the `MultiFileCubeThreshold` setting in the cogtr.xml file to enable creation of multiple local PowerCube files. This multiple-file cube has one file with the .mdc extension and one or more files with the .mdp (multi-dimension partition) extension. To PowerCube consumers, multiple-file cubes appear as a single large cube.

The default `MultiFileCubeThreshold` value is 0, which signifies that multifile generation is disabled. However, you can change this threshold by adjusting the `MultiFileCubeThreshold` setting value.

Note: Transformer determines the number of output files needed by taking the number of data records in the cube, dividing by the threshold, and rounding up.

Multifile cubes must be partitioned. The contents of the partitions are evenly distributed across the required number of multidimensional partition (.mdp) files, and an additional multidimensional cube (.mdc) file is added to hold the PowerCube metadata. Each file cannot exceed 2 GB.

For more information about configuring Transformer, see the *Installation Guide* or "cogtr.xml File Settings" (p. 347).

Recover a Failed Model

To help you recover from power failures or similar processing interruptions without having to recreate the entire cube, Transformer adds checkpoint entries after each major stage in the cube creation process. These checkpoints are written to a temporary .qy? file in the folder specified by **Model temporary files** on the **Directories** tab of the **Preferences** property sheet. The .qy? file is deleted when processes end normally; as a result, the existence of a .qy? file indicates that Transformer terminated unexpectedly.

The next time you start Transformer, you will be asked if you want to see the list of suspended models. You can either open the model at the last checkpoint before failure and continue to develop it from that point, or continue from the point where you last saved the file. The .qy? file is deleted.

Transformer also writes messages to a log file which is stored in the location specified on the **Directories** tab of the **Preferences** property sheet. The log file has the same name as your model, with a .log extension. If Transformer is unable to recover automatically from the processing failure, or if you choose to ignore previous processing and begin again, you can read the log file to find and correct the cause of the failure.

Note: Recovery from a checkpoint file is not supported if auto-partitioning is enabled.

Steps for the Windows Interface

1. From the **File** menu, click **View Suspended Models**.

   Note: If a severe error occurred during processing that caused Transformer to shut down, you are prompted to open the **Select Suspended Model** dialog box when you next start Transformer.

2. Select the option to open a model and click **OK**.

Steps from the UNIX or Linux Command Line

1. To open a checkpoint file in batch mode, use the `-p` command line option; for more information, see "Command Line Options" (p. 237).

2. To ignore a checkpoint file in batch mode, use the `-i` command line option; for more information, see "Command Line Options" (p. 237).

Match Model and Source Columns

When you create a model, the columns in each data source are saved as part of the model definition. If you reorder, add, delete, or rename the columns in any data source, you must update the model by running the **Modify Columns** command.

For example, suppose your source file has columns named `DATE`, `PRODUCT`, `SALESPERSON`, `QTY_SOLD`, and `PRICE`. The sales representatives are reorganized into units, and a `UNIT` column is added to store this new transactional data. You run the **Modify Columns** command to update your model.

For more information about the **Modify Columns** feature, see "Synchronize Columns with Your Data Source" (p. 62).

Steps for Text Data Files or IQDs

1. In the **Data Sources** list, select the query whose columns you want to modify, and from the **Tools** menu, click **Modify Columns**.
2. For data source items that do not appear as columns in the model, select the items in the **Source** list and click **Add**.
3. For columns that you want to remove from the model, select the columns in the **Model** list and click **Remove**.
4. For unmatched columns identified by a plus sign (+) in the **Matched to Source** column in the **Model** list, do one of the following:
   - To manually match query items to columns, select a data source item in the **Source** list and a column in the model, and click **Match**.
   - To allow Transformer to automatically match query items to columns, click **Auto Match**. Review any messages that appear and click **OK**.

Data source items are now synchronized with the model.

Steps for IBM Cognos 8 Data Sources

1. In the **Data Sources** list, select the query whose columns you want to modify, and from the **Tools** menu, click **Modify Columns**.
2. If there are columns in the model that cannot be matched to data source items, you will get a warning message. Click **No** to keep these unmatched columns in the model.

   If you click **Yes**, Transformer deletes the unmatched columns.
3. For data source items that do not appear as columns in the model, select the items in the **Source** list and click **Add**.

   Tip: Click **Refresh Source** to refresh the source list for the data source. Click **Validate** to check whether columns in the model violate any Framework Manager governor settings.
4. For columns that you want to remove from the model, select the columns in the model list and click **Remove**.

5. For unmatched columns identified by and **X** in the **Matched to Source** column in the model list, do one of the following:

   - To manually match query items to columns, select a data source item in the **Source** list and a column in the model, and click **Match**.

   - To allow Transformer to automatically match query items to columns, select the columns in the model and click **Auto Match**.

6. If Transformer provides one or more locations in the data source that may be appropriate for unmatched columns, do one of the following:

   - If one of the locations is an appropriate match, select the location and click **Next** or **Finish**.

   - If none of the locations are an appropriate match, click **Match by reference instead** and click **Next**. From the **Candidates** list of data source items, click the one that you want to match to the column or click **Leave unmatched**.

7. Repeat step 6 for each mismatched column that Transformer attempts to resolve.

8. For any mismatched items that Transformer cannot suggest locations for, Transformer presents possible name changes for your review. From the **Candidates** list, click the item that you want to match to the column to or click **Leave unmatched**.

Data source items are now synchronized with the model.

Move Categories When Source Data Changes

You can prevent model errors due to changes in your source by designating a unique level. This tells Transformer that categories in that level are identified by their source values alone, without reference to their ancestors. However, when you move a category within a unique level, a uniqueness violation is reported during category generation because the moved category now appears in a different context.

To avoid having to manually restructure the categories in a unique level to conform to the changed ancestor data, you can specify that such changes be treated as unique moves. Measure values are thereafter rolled up the new path from the moved categories.

If any cube in the model is set to incrementally update, you cannot specify changes as unique moves. Conversely, if unique moves are specified for any level, you cannot set incremental updates for the cubes.

Unique move is supported for time-based partitioned cubes. In the member cubes that were created before the category was moved, the category will remain connected to the previous parent and values will roll up to that parent. In member cubes that were created after the category was moved, the category will be connected to the new parent and values will roll up to the new parent.

In IBM Cognos 8, any categories that do not move under a different parent will have the effect of a changed member unique name for that category. When reports that directly reference this category or member are run, the report consumer will receive an error. The report author will need to reassociate the category, or member, to the newer version.

For more information about member unique names, see "Member Unique Names" (p. 45).

Steps

1. Open the property sheet for the level that contains the categories affected by the changed data.
2. On the **Source** tab, click **Unique**.
3. In response to the warning message that categories in the level must be recognizable by the source data alone, click **Yes**.
4. Click **Move** and click **OK**.
5. From the **Run** menu, click **Generate Categories**.
6. To check that the categories moved correctly, select the dimension in the dimension map and click **Show Diagram** from the **Diagram** menu. Or verify that measure values rolled up correctly by viewing the results in one of the OLAP reporting components.

Example - Moving Categories When Source Data Changes

You must take special care to avoid inaccurate measure data in your reports when categories in your model have moved from one location to another.

Suppose your organization has two staff members - Alessandra Torta in Milan (Employee Number 5528) and Ellen Shapiro in Amsterdam (Employee Number 4125). After these two employees switch offices, you want the measure data associated with them to move accordingly.

If you simply make the switch in your source data without specifying that the Employee Number level is a unique level, Transformer generates two categories for each staff member: the category under the former office and the category under the current office. The measure values associated with each individual remain associated with their former offices.

The correct procedure is to specify that the employee number level is a unique level, and the switch in location is a unique move, by clicking both the **Unique** and **Move** check boxes in the **Level** property sheet for the employee number.

Your users now see the correct results in their OLAP reports; each staff member category is moved to the new office.

Recommendation - Periodically Clean up Your Models and Cubes

As you implement larger and more complex models and cubes, optimizing build times and run-time performance becomes increasingly important. To maintain peak performance, we recommend that you periodically reassess your model design.

To speed up cube creation, reduce cube size, or improve run-time access for your OLAP report users, you may need to

- adjust your data
- streamline your data sources
- redesign your model

- select different cube processing options

Tip: When you make changes to your model, be aware that the changes you make may impact report authors and report consumers. For more information, see "Member Unique Names" (p. 45).

Consider making the following improvements:

- Delete records from the source data if they are no longer needed or are out of date. Consider excluding data from the cube, or consolidating the data by summarizing the details.

- Use multiple data sources where feasible, to reduce the size of each source file and shorten processing time in Transformer. For more information, see "Use Multiple Data Sources" (p. 200).

- Subdivide large PowerCubes into multifile cubes to improve run-time query performance.

- Ensure that structural data sources (those that contain the hierarchical data for your dimensions) appear first in the **Data Sources** list, followed by the transactional data sources (those that supply the measures for your model). Include only the minimum number of columns needed to reference your dimensions.

- If you specify a unique level, ensure that your data does not have uniqueness violations. Allocate extra time for data source processing to verify that all categories are unique within a level, or eliminate this step if it is not necessary by clearing the **Verify category uniqueness** option on the property sheet for the data source.

- Consolidate transactional data to a higher summary level using the **Duplicates Rollup** feature, and implement optional drill-through targets if the details are not needed by the majority of your users.

You can also select one or more of the following options from the **Processing** tab of the **PowerCube** property sheet.

Optimization

Use this option to specify how cube creation is optimized. Auto-partition is the preferred optimization method.

The optimization methods include:

- Auto-partition

  Auto-partition is the default, and recommended, optimization method. With Auto-partition, you can let Transformer determine where to place the partitions or you can define the partitions manually.

  Some situations may prevent the use of Auto-partition optimization. In these situations, Transformer uses Categories optimization. For example, if a model has two cubes, one with optimization set to Auto-partition and the other with optimization set to Categories, Transformer will build both cubes using the Categories method. This is because Transformer cannot use two different optimization methods at the same time. The **Check Model** feature issues a warning indicating when Categories optimization is used instead of Auto-partition.

  For information about cube partitions, see "Choosing a Partitioning Strategy" (p. 203) and "Partition Manually (if Required)" (p. 206).

- Categories

  Categories is an older optimization method. When cubes are incrementally updated, Transformer uses the Categories method.

  The Categories method can handle certain model conditions, such as a before-rollup calculated measure or selecting no consolidation when processing records for a cube, but this is not the recommended optimization method.

  If you want to partition the cube and use the Categories optimization method, you must set the partitions manually in the cube. In addition, more than one level of partitioning will increase the cube build time substantially.

- Data passes

  The data passes optimization method is a variation of the categories method. When Transformer uses the data passes method, the number of passes through the temporary working files created during cube creation is reduced. All categories are placed in the cube, however those with no measure data attached are hidden when the cube is read.

  With the data passes optimization method, the resulting cube may be larger and cube read processing may be slower. This is because duplicate data points are not consolidated, and all categories are included in the cube.

- Direct create

  The direct create optimization method is also a variation of the Categories method. When Transformer uses the direct create method, all categories in the model are added to the cube while the data sources are being processed. Records that do not generate new categories are directly updated to the cube.

  We recommend that you use this method only when your model is expected to generate few new categories, and when all categories are added to the cube.

  You cannot use the direct create method for individual PowerCubes in a cube group.

This Cube is Incrementally Updated

Use this option to add new data to an existing cube. If your records change only slightly, this option reduces the time needed for cube processing.

Alternatively, you can create data sources that contain only the new data, and use these sources to incrementally update your cube.

Remember to keep your model synchronized with your incrementally updated cube and, even if your model does not change, recreate your cube periodically in its entirety. If you do not recreate your cube periodically, and both the frequency and number of incremental updates is high, processing performance may deteriorate over time. For more information, see "Update Cubes Incrementally" (p. 202).

Cube Creation

Use this option to create the cube or cube group locally. The default is **Enabled**. If data related to a particular cube is unchanged since the last update, shorten processing time by selecting **Disabled**.

For time-based cubes, child cubes are always published with the time-based cube. Disabling publishing for a child cube only restricts publishing of that cube individually.

Enable Crosstab Caching

Use this option to store summaries for the initial crosstab in the cube. These summaries help improve run-time access for your OLAP report users.

Block Totals for Parents with Excluded Children

This option controls whether values are displayed against categories that are the rollup ancestors of categories that have been excluded in a custom view. When this option is selected, an ancestor category displays **denied** instead of a value. To see values, users must drill down to categories without excluded descendants.

Use Multiple Data Sources

By using multiple data sources for your model, you can reduce the total volume of data processed and thereby shorten processing times in Transformer. If used in combination with calculated columns, multiple data sources can minimize or eliminate the need to create database table joins in an external data access tool. Using multiple data sources also enables measure allocation.

For example, suppose your product, customer, and order data is stored in a set of tables. If you were to use this data from a single source, you would need separate tables for Product, Customer, CustomerSite, Order, and OrderDetail. This source would contain many duplicate values, and the joins between the tables would be relatively complex.

Instead, you create three separate sources for Products, Customer/Site, and Order/Order Detail data. The volume of data contained in each is less than that in the single source, and there are only simple joins between Customer and CustomerSite tables, and Order and OrderDetail.

Remember that each source must contain sufficient data to establish context within the dimension map. You cannot perform database table joins in Transformer.

Steps

1. Using a data access tool such as IBM Cognos Impromptu or Framework Manager, create each of the data sources required for your model.

2. From the Welcome page, click **Create a new model** to use the **New Model** wizard to add the largest structural data source to your model.

   Tip: If you are already in Transformer, click **New** from the **File** menu.

3. From the **Edit** menu, click **Insert Data Source** and add the additional structural data sources to the **Data Sources** list.

4. Repeat to add the transactional data sources to the **Data Sources** list.

Consolidate Data in PowerCubes

In Transformer, consolidation summarizes measure values during cube creation, so that fewer summarizing calculations are required in the OLAP reporting component. Consolidation reduces cube size and, if you enable auto-partitioning, does not slow down cube creation.

Tip: Because the consolidation process runs against every source file in your model, ensure that your temporary file space is at least as large as the sum of all your source files.

Steps

1. Open the **PowerCube** property sheet for the cube that you want to consolidate and click the **General** tab.

2. In the **Consolidate** box, click the appropriate entry in the drop-down list:

   - **Default**

   - **No**

   - **Yes (with sort)**

   - **Yes (presort)**

3. Click **OK.**

Disable the Category Refresh Option

For large complex models, you may improve processing by disabling the options that refresh the category information.

If this information subsequently changes in your source data, remember to enable the refresh options again before regenerating your categories.

Steps

1. Open the appropriate **Level** property sheet so that you can disable its refresh options.

2. Clear the **Refresh** check boxes for **Label**, **Description**, and **Short Name**.

3. Click **OK.**

Delete Inactive Categories

If your organization changes, you can delete inactive categories that are no longer needed in your model.

A category is only considered active if it has been created, updated, moved, or had its properties changed after the date you specify.

For example, suppose that last year your department was responsible for inspecting 40 homes for the elderly. After a consolidation of facilities, you now inspect and report on the operations of only 30 homes. You delete the categories for the 10 homes that have been closed since the start of your fiscal year.

The **Clean House** feature should be used with caution, as it may cause problems with consumers' reports in IBM Cognos 8. If you intend to delete inactive categories, we recommend that you advise your consumers so that they can remove those categories, or members, from their reports when they are directly referenced. Otherwise, they may receive an error the next time they run their reports.

You cannot delete inactive categories if incremental updates are defined for the cubes in your model. If you do, an error message will appear when you try to create the cube.

Steps

1. From the **Tools** menu, click **Clean House**.
2. In the **Date** box, type a date using the format configured for your system or select a date by using the embedded calendar, and click **OK**.

   This action removes all categories not created or changed since your specified date.
3. Regenerate the cube and confirm that the stale-dated categories are no longer present.

Modifying a PowerCube

If the business information requirements or data access rights of your report users change, you will likely need to make corresponding changes to your PowerCubes or cube groups.

Update an Existing PowerCube

If your changes are minor, you can use the **Update PowerCube** feature (**Run** menu), instead of deleting and replacing the entire cube, to update

- object names, short names, labels, and descriptions
- security objects and their associated views
- currency conversion tables

Omit Dimensions Used Only for Detail Reports

If your model has partitioning on detail dimensions, querying at the lowest (leaf) level will be slow because reporting must be done across multiple partitions. To avoid this problem, you can create a separate PowerCube that omits dimensions used only for lowest-level detail reports. The resulting cube is smaller, quicker to build, uses smaller partitions, and processes summary queries more quickly.

You can choose to omit dimensions from a cube, even if it uses auto-partitioning.

For more information, see "Omit Dimensions from a Cube" (p. 158).

Update Cubes Incrementally

It is best to place frequently updated data in one or more separate data sources, so that you can perform incremental updates to quickly add the latest changes to your existing cube.

Tip: Make sure that your data sources contain only new data; otherwise, data duplicated in your source files will be duplicated in the cube.

Incremental updates can save processing time. However, you must consider the following:

- The first time you run an incremental update, include all the categories for the levels used for partitioning. This ensures the best performance from an incrementally updated cube.

- If you add new categories to a partition level in a model, the new categories are stored in the parent partition level.
- If you add a large number of new categories, the access time for report users may be adversely affected because of the increased size of the summary cube. Also, over time, if new categories are introduced, the incremental updates may reduce the effectiveness of the partitioning scheme in the cube, which degrades performance. In either of these cases, we recommend that you repartition your model.
- After the first incremental update, Transformer reverts to using the less-effective **Categories** method, rather than the **Auto-partitioning** method.

Tip: If your model uses cube groups, and only a subset of them contain data that has changed since the last update, you can save processing time by disabling the cube creation option for the unchanged member cubes.

For performance reasons, we recommend that you periodically recreate the entire cube, to optimize its auto-partitioning scheme (for example, update weekly, recreate monthly). Also, if you make structural changes to your model, you must recreate the cube, including its historical summary data, before you can perform further incremental updates. We therefore recommend that you retain all the data sources used to create the original cube, so you can point to those sources during the rebuild.

Constraints

To avoid invalidating your incrementally updated cubes

- do not delete a leaf category, or use **Exclude**, **Cloak**, **Summarize**, or **Apex** in a view, if this would cause leaf categories to be removed from the cube
- do not delete, add, or move a dimension or measure

Steps

1. Add a new cube definition to the **PowerCubes** list.
2. Open the property sheet for the new cube and click the **Processing** tab.
3. Select the **This cube is incrementally updated** check box and click **OK**.
4. Create the cube.

Choosing a Partitioning Strategy

Partitioning is an effective way to improve run-time access to cubes and OLAP reports that have millions of rows of source data. There is a tradeoff, however. The more partitions there are, the longer it takes to build the cube.

The partitioning process involves multiple passes through the data. Categories are consolidated and sorted, records in the resulting (temporary) work files are counted, and the information is divided into rows that summarize the measures at one or more optimal partition levels.

By default, Transformer automatically partitions your cubes. However, if your cubes are very large or unusually structured, or if you have particular reporting needs, you may want to designate one or more dimensions for manual partitioning, provided these dimensions have sufficient depth. Flat hierarchies do not lend themselves to this optimization method, although you can add manual levels to bring them closer to the ideal parent:child ratio (1 parent to 10 children).

Partitioning is warranted, and will most likely be successful, in the following circumstances:

- Most of your users' queries can be answered from the first or upper partitions, called summary partitions.

  If you partition a dimension for which lowest-level detail reports are needed, access times will be slower, as the data must be returned from several partitions.

- The information required for most queries can be found within a single partition.

- Your dimensions have sufficient depth that the category-per-level ratios are 10:1 or less.

  If a dimensional hierarchy is too flat, you may have to add manual levels to bring it closer to the ideal ratio.

Constraints

In developing your partitioning strategy, consider the following constraints:

- You cannot use auto-partitioning if **Consolidate** is disabled on the **General** tab of the **PowerCube** property sheet, or if **Optimization** is set to any method other than **Auto-Partition** or **Default**, such as **Categories** or **Direct create**.

- Auto-partitioning is disabled if your model has a custom view with a cloaked primary-drill category. You can run **Check Model** to receive a warning message about this condition, allowing you to uncloak the drill category or otherwise redesign your model to allow auto-partitioning to proceed.

- Although auto-partitioning may occasionally partition categories from dimensions that have alternate drill-down paths, you can only use manual partitions on categories in the primary drill-down path.

- You cannot auto-partition cubes that use externally rolled-up or before-rollup calculated measures. However, you can partition cubes that use calculated columns.

- You should avoid partitioning dimensions that are frequently updated with new categories, or that contain alternate drill-down structures, leaf-level subdimensions, or special categories.

- You cannot specify partition numbers for leaf categories, drill categories, or the main root category.

- You can add new categories to an existing partition level. For example, you can add a new region, which will result in a new partition for that region. However, you cannot add new partition levels to a model if any cube in that model uses incremental update. Instead, you must repartition the model and rebuild the cube, incorporating all the data from all increments.

Partitioning Checklist

When partitioning, confirm the following:

❑ Ensure that you set up clearly recognized, business-related groupings in your source data, such as commodity types or customer sales channels, and import them into your model as a new data source column.

❑ Because you cannot set different partition sizes for different parts of the cube, consider reorganizing your data so that some cubes can be optimized for high-level summary reports and others for low-level detail reports.

❑ Add calculated columns, such as product number ranges or alphabetically sorted customer sets, to regroup your data.

❑ On the dimension map, manually create alternate drill downs with manual levels and create categories representing subdivisions, such as geographic regions, and use the category viewer to drag lower-level categories to associate them with the appropriate parent categories.

Tip: If you do not need these manually added categories in every cube, create a dimension view and use the **Suppress** command to remove the unnecessary levels of detail. However, suppressed categories cannot be partitioned and suppressing the manual levels in a custom view will reduce the effectiveness of your overall partitioning strategy.

For more information about suppressing categories, see "Omit Categories Using Suppress" (p. 161).

Adjust the Auto-Partitioning Parameters

You can use the **Auto-Partition** tab on the **PowerCube** property sheet for each cube in your model to test different settings and see which ones yield the best results for your situation.

Before you begin, ensure that any previously defined manual partitions are cleared, and that the optimization setting on the **Processing** tab of the **PowerCube** property sheet is set to **Auto-partition**.

Steps

1. Click the **Auto-Partition** tab and experiment with various test settings, as follows:

   - Specify an **Estimated number of consolidated records**. Consolidation combines records that contain identical non-measure values, reducing cube size and improving run-time performance. Start with the default. Later, you can check the log file to see the actual number.

   - Adjust the slider in the direction of **Faster cube creation** or **Faster cube access**, as appropriate for your current requirements. By default, the slider is centered.

   - Specify a **Desired partition size**, which is based on the consolidated record count. The default number of categories is 500,000 but you can reduce this number in incremental steps, testing various settings until you achieve acceptable run-time performance. When you build your cube, Transformer automatically groups all categories into partitions of this size, in a way that is optimal for your model. Partitioning stops when the number of records in the summary partition drops below this specified partition size.

- Specify the **Maximum number of passes**, beginning with either five (`5`), or one (`1`) pass for every manually defined partition level. The partitioning process continues, one pass per level, until this number is reached, or until subsequent consolidation passes fail to yield a further reduction in the summary partition size.

  Tip: Set this value to the minimum number of passes needed. If tests show that the last few passes did not significantly improve performance, decrease the number to achieve an appropriate balance between data access and cube build times.

2. Click the **Dimensions** tab, select the dimensions that you want to include in your cube, and click **OK**.

3. Click **PowerCube Partition Status** on the **Tools** menu, learn how the automatic partitioning algorithm has distributed your records during your partitioning tests, and act on the results. For example, if both your summary partition and your level-one partition have the same number of records, then the summary partition has not been consolidated. You need to increase the number of passes.

   Tip: You may want to perform some manual partitioning tests, comparing those results with the performance gained by auto-partitioning and choosing the best solution for your particular OLAP reporting situation.

Partition Manually (if Required)

If your model design does not support auto-partitioning, or if you have a model with an unusual structure, you may achieve better performance by setting up the partitions manually.

Steps

1. Let Transformer auto-partition your cube, leaving the optimization settings at their defaults.

2. In the resulting log file, note the partition points selected by the auto-partitioning algorithm.

   Generally, the dimensions with the most levels offer the greatest possibility for consolidation. Review the difference between the row and category counts in the **Start Count and Consolidation** and **End Count and Consolidation** lines, comparing records consolidated between `Pass 0` and the last pass (usually `Pass 4` or `Pass 5`).

3. Remove partitioning from your model so you can start testing your manual alternatives, by clicking **Reset Partitions** on the **Tools** menu.

4. Calculate the required number of partitions, based on the following formula:

   Number_of_partitions=Number_of_source_rows/Desired_partition_size

   Note: In most production environments, partition levels should not exceed `3` and partition sizes should not exceed `500,000` records. Source rows are based on the source file.

5. Select a dimension suitable for partitioning. Avoid large flat dimensions, dimensions with alternate drill-down paths or leaf-level reporting and dimensions whose categories change frequently, such as the time dimension. Open the diagram for the selected dimension.

6. Choose a level that contains the approximate number of categories identified in your calculation, and open the property sheet for each category in that partition root level.

7. In each **Partition** box, type the partition number you want to assign and click **OK**.

   Note: Although you can assign the same partition number to categories from different levels, we recommend that you select categories from the same level, and manually set no more than three levels of partitioning.

8. Assign the same number to each category in the partition level, starting with the lowest level and working up to the top level in the dimension.

   Make sure that each partition level number is equal to, or one greater than, the partition number of its ancestor. You are warned of any gaps in the numerical order of the partitions when you create the cube.

   Note: Manually-specified partition level numbers do not necessarily define the order in which partitions are placed in the cube. Partition level numbers may be reordered by Transformer.

9. On the **PowerCube** property sheet, click the **Auto-Partition** tab and specify a **Maximum number of passes** that is at least as large as the number of partitions you added in the previous step.

10. Create the cube, analyze the log file to see if the number of records and categories decreases from pass to pass and, if there is no decrease in the number of records on the last pass, reduce the **Maximum number of passes** value by one, or increase the desired partition size. If both the summary and `Level 1` partitions have the same number of records, indicating that the summary partition was not consolidated, try increasing the **Maximum number of passes**.

11. Use the **PowerCube Partition Status** command on the **Tools** menu to view the distribution of records. For each applied partition level (beginning with `0`), a window shows the dimension name, the category code, and its record count. Check to see if any of the partitions are larger than your specified partition size.

12. Open the cube in your OLAP reporting component and try drilling down to the partition with the largest number of records. If performance is unacceptable, reset your partitions and add another level of partitioning to the dimension you chose. Alternatively, select a different dimension where the gains may be more favorable. For each new level of partitioning, remember to increase the partition number by one (`1`).

Example - Partitioning Manually When Auto-Partitioning Cannot Be Used

Partitioning can be difficult. If your manual optimization efforts are not effective, you may have to ask an expert for assistance.

Suppose you have a flat-dimensioned model consisting of product IDs, customer numbers, and countries. The ideal parent:child ratio of 1:10 is exceeded; there is only one parent for each of the large child category levels (Product Brand, Customer Type, and Region).

Your customers need summary reports on categories from the Product and Customer dimensions, so you manually partition at a higher level in those dimensions. You specify which levels and categories are included in the partitioning and set the **Maximum number of passes** on the **Auto-partition** tab to that number of partition levels plus one.

However on testing, you discover that additional partitions are needed. You try various strategies, using the Products dimension.

First, you assign a single partition to the entire Product Type level, which results in the following:

```
Level 0, All dimensions, <Summary> category - record count of 237
Level 1, Line dimension, <Type 1> category - record count of 31
Level 1, Line dimension, < Type 2> category - record count of 181
Level 1, Line dimension, < Type 3> category - record count of 25
```

However, drill-down performance is only acceptable when querying Types 1 and 2. It is unacceptably slow for Type 3, because there are too many records in the partition.

To more evenly distribute the total number of categories, you navigate to the **General** tab of the **Category** property sheet for each product type, and assign the same partition number (`1`) to all Type 3 child categories, and to all Type 1 and 2 categories at the parent level.

You rebuild the cube, check the partition status, and confirm that the Product Type category counts are better balanced, with no partitions containing more than 87 records. However, you note that some Region categories contain a large number of child records.

You decide to add individual Country categories to the same partition as that assigned to the parent, to further optimize partitioning. You then recheck query performance, but conclude it is still unsatisfactory.

Having reached the extent of your own expertise, you gather the following data and prepare to contact Cognos Software Services, to have them investigate your problem further:

- the Transformer log file from the PowerCube build, with information about which dimensions were involved in the partitioning, and partition consolidation details
- the populated Transformer model (.mdl or .py? file) and PowerCube (.mdc or .mdl file), so that IBM Cognos support staff can confirm the status and record distribution of all partitioned categories
- information about which dimensions are most queried at run-time, and how deep those queries tend to be; that is, whether your users query mostly the parent or child levels
- detailed information about cases when query performance was below expectations, such as typical reports and OLAP activity, especially regarding nesting or alternate drill-through exploration

Recommendation - Keep Upgraded Models Populated if Category Codes are Not Unique

When the category codes, or members, in Transformer models are not unique, Transformer version 8.x makes them unique by adding a numeric sequence to each code, preceded by a tilde character (~). This blended expression cannot be interpreted by other IBM Cognos 8 components.

For example, suppose your cube contains more than one instance of the category code **Item**, and so renames the second instance **Item~1**, the third instance **Item~2**, and so on. These category codes are numbered as they are encountered in the model. This means that the codes do not stay the same, in a predictable sequence, as your organization (and model) evolves. The tilde characters are interpreted as non-numeric and, because these codes do not appear in the data source, they are ignored during the consolidation pass. As a result, duplicate records are overridden and not summed, despite what may be reported in the log file.

If the category codes are not unique, and change after each cube build, reports and drill-through parameters that contain explicit member references may no longer run successfully. To avoid this problem, we recommend that you base your reports on level (children of) references, which are not affected by category code changes.

We recommend that you save your models with categories of this type in their fully populated form, rather than cleaning out the categories between each cube build. We also recommend that you save the models as both .mdl and .py? files, so that you have a backup version if the .py? file becomes corrupted. This precaution is particularly important if the same category repeats itself, or if your model contains blank categories.

To avoid such problems, use one of the following strategies to make these codes unique:

- Ensure that all source values are unique in a dimension.
- Create a calculated column in Transformer, to make your categories unique.
- Edit the Model Definition Language (MDL) model file to make the category codes unique.

  For more information about using MDL, see the Transformer *Developer Guide*.

Set up Placeholder Categories for Cube Groups

When you define a cube group, you can manually add a placeholder category to the level from which future cubes will be derived to allow for additions. Provided the new members have the same rolled-up data as the existing members, the cube containing the placeholder category will gather the incrementally updated data that the new member needs.

For example, suppose a model uses a cube group for your sales offices, which you update incrementally. Each member cube therefore contains historical summary data. Your company is planning to add another sales office in the future, but details are not yet available. You want the new office to have access to the same historical data as the other offices.

Some limitations apply. For more information, see "Update Cubes Incrementally" (p. 202) and "Choosing a Partitioning Strategy" (p. 203).

Steps to Set Up a Placeholder Category and Cube

1. Open the diagram for the dimension on which the cube group will be based.
2. In the category viewer, insert a category into the level from which the member cube will be derived.
3. Define the cube group and, on the **Processing** tab of the **PowerCube** property sheet, select the **This cube is incrementally updated** check box.
4. Click **OK**.
5. In the **PowerCubes** list, expand the cube group to list the members.
6. In the property sheet for the placeholder cube, change the name to **New**.

Steps to Set Up the New Cube Group Member

1. Open the diagram for the dimension on which the cube group is based.

2. On the property sheet for the placeholder category, rename the **Category Label** and **Source file** to match the new data column.

3. On the property sheet for the placeholder cube, rename the cube.

Update Published PowerCubes and PowerCube Connections

After a PowerCube is rebuilt or updated, several methods are available for making the newer version of the published cube available. You can

- copy the newer version of the cube to the deployment location(s) of the existing version, and then activate the newer version.

  This is the recommended method for making newer versions of published cubes available. This method is fully automated, which reduces the risk of errors that may cause connection problems when users attempt to access the cubes.

- update the data source connection information to point to the newer version of the cube using the IBM Cognos 8 PowerCube Connection Utility. This command line utility is located in the *installation_location*/c8/webapps/utilities/PCConn directory.

  Note: The IBM Cognos 8 PowerCube Connection Utility will be deprecated in the next release of IBM Cognos 8 Transformer.

- disable the published PowerCube using the IBM Cognos 8 PowerCube Connection Utility so that the existing cube can be overwritten with a newer version.

To use the IBM Cognos 8 PowerCube Connection Utility to update a data source or to disable a cube, you must first connect to a dispatcher. You can connect to multiple dispatchers if your IBM Cognos 8 environment requires it.

If your IBM Cognos 8 environment is secured, you are prompted to provide authentication information when you connect to the dispatcher. If your environment has multiple namespaces, you can log on to any or all namespaces.

Copy and Activate a Newer Version of a Published PowerCube

After rebuilding or updating a PowerCube that has already been published, you can make the newer version of the PowerCube available without interruption to IBM Cognos 8 users. Transformer can copy the newer version to one or more deployment locations that you specify, and then activate the newer version in each location. Reports continue to run using the new cube data.

You can initiate the copy and activate process using Transformer commands or from the command line(p. 213).

Tip: You can also use pcactivate to activate updated Transformer 7.x PowerCubes already copied to the production environment (p. 213).

To copy and activate a PowerCube, Transformer does the following for each deployment location:

- If this is the first time the cube has been copied and activated, Transformer creates a small file named *cube_name*.mdc that represents the PowerCube.

- Transformer creates a folder named *cube_name__UTCdate_time* and copies the new version of the cube to this folder.

  UTCdate_time represents the date and time that the copy process started, in Coordinated Universal Time. For example, the following represents March 12, 2008 at 5:58:39 p.m. in UTC format: 20080312175839. This time format ensures that folder names represent a standard global time, regardless of the time zone in which cubes were copied and activated.

  For time-based cubes, this folder contains additional contents, including a control cube, time-based control file (.vcd), and member cubes.

- If a *cube_name__UTCdate_time*.ver file exists, Transformer deletes this file. Only one *cube_name__UTCdate_time*.ver file can exist in each deployment location.

  Transformer creates a file named *cube_name__UTCdate_time*.ver. The prefix of this file name (*cube_name__UTCdate_time*) is the same as the folder in this directory that contains the newest version of the cube.

- Transformer writes messages about this process to a log file, which is stored in the location specified on the Directories tab of the Preferences property sheet.

For a cube named MyCube.mdc, where one version was copied and activated on February 1, 2008 and a newer version was copied and activated on March 1, 2008, a deployment location contains contents similar to the following:

- a file named MyCube.mdc
- a file named MyCube__20080301175839.ver
- a folder named MyCube__20080301175839 that contains the version of MyCube.mdc that was copied and activated on March 1
- a folder named MyCube__20080201125819 that contains the version of MyCube.mdc that was copied and activated on February 1

Assume that the report server needs to process a query using MyCube.mdc. When attempting to access MyCube.mdc, the report server recognizes that the file is not a valid PowerCube. The report server looks for a file named MyCube__*UTCdate_time*.ver file. In the previous example, this file is named MyCube__20080301175839.ver. Using the prefix of this .ver file, the report server goes to the MyCube_20080312175839 folder and processes the query using the version of MyCube.mdc in this folder.

You can use a different value than *UTCdate_time* if you manually change this value in the deployment location. Ensure that you change the value in both the prefix of the .ver file and the folder that contains the newest version of the cube. The format for both the prefix and the folder name must include two values that are separated by a two underscores (__), as follows: *value__value*. For example, for the .ver file, you can use the following: MyCube__1v2.ver. You cannot use MyCube_1v2.ver (single underscore) or MyCube##1v2.ver (two hash symbols).

Copying and activating newer versions of cubes should be used only when updated data is available.

Step

- In the **PowerCubes** list, right-click the PowerCube and click **Copy and Activate Selected PowerCube**.

  Tip: If you created new versions of more than one PowerCube, you can copy and activate them at the same time using **Copy and Activate PowerCubes** from the **Run** menu.

Steps to Set Up a Copy and Activate Strategy

1. In the **PowerCubes** list, right-click a PowerCube and click **Properties**.

2. Click the **Deployment** tab.

3. From the **Deployment Strategy** list, select the deployment method:

   - Select **Copy to available locations, then activate** to copy and activate the PowerCube in all specified deployment locations.

   - Select **Copy only if all locations are available, then activate** to copy and activate the PowerCube only if all deployment locations are available.

     If one or more of the deployment locations are unavailable, the deployment action is aborted for all specified locations.

   Tip: If the cube is a child cube belonging to a cube group and you want the child to inherit the same deployment strategy as its parent cube, click **Use Strategy from Parent Cube** and go to Step 6. This option is not available for standalone cubes or for member cubes of time-based partitioned cubes.

4. From the **After building the cube** list, choose the copy and activation strategy to be applied each time the cube is rebuilt:

   - Select **Prompt to copy and activate** to have Transformer prompt for confirmation before deploying and activating the rebuilt cube.

   - Select **Automatically copy and activate** to have Transformer copy and activate the cube automatically each time it is rebuilt.

   When you select **Do not deploy**, Transformer performs no other action when the cube is rebuilt.

5. Click **Add**, in the **Folder** box, enter the path to the cube, and then click **OK**.

   Tip: To enter the path to the cube, you can go to the location where the cube is deployed in the **Deployment Location** pane; type the path to the cube in the **Folder** box; or copy the path to the cube and paste it in the **Folder** box.

   If this PowerCube is available from multiple locations, add each location to the **Deployment locations** list.

6. To enable automatic deletion of previous versions of the deployed PowerCube, select the **Enable automatic PowerCube deletion** check box and then, in the **Maximum number of old cubes to keep** box, type the number of previous versions to be retained.

7. Click **OK**.

A message confirms that the deployment was successful. View the message details for the name of the folder in which the newer version of the PowerCube is stored.

The newer version of the PowerCube has been copied to all deployment locations and activated. In each location, the .ver file is updated to identify the name of the folder in this directory that contains the activated PowerCube.

Activate a Published PowerCube Using PCactivate

Use the `pcactivate` command to activate PowerCubes that are rebuilt or updated in the production environment. This is the recommended method for making newer versions of the published cube available. This method is fully automated, which reduces the risk of errors that may cause connection problems when users attempt to access the cubes. You can use the `pcactivate` command with updated Transformer version 7.x cubes and updated Transformer version 8.x cubes.

The `pcactivate` command:

- renames the directory in which the updated PowerCube is located to include a version reference number
- creates a version (.ver) file that directs the Web studios to access the updated version of the cube
- deletes all other .ver files for the cube

The next time the published cube is accessed, the newer version is accessed automatically.

Before you can use the `pcactivate` command, the cube must be

- copied to a directory in the production environment with the same name as the cube

  If the cube is in multiple parts or is a time-based partitioned cube, all parts must be copied to destinations that are in the same location relative to the base cube.

- properly located relative to where the cube appears to be, as defined in the data source definition in Content Manager

  It must be in the same directory where IBM Cognos 8 will look when opening the PowerCube.

For example, if the data source definition indicates that the access path for the cube is c:\Cubes\Product.mdc, you must copy the cube as indicated below before you can use the `pcactivate` command:

c:\Cubes\Product\Product.mdc

The syntax for using the `pcactivate` command is as follows:

```
pcactivate cube_name.mdc [destination_location] [destination_location]
```

If you type a path in the *cube_name.mdc* parameter, it is ignored.

Note: You cannot run the pcactivate executable, pcactivate.exe, outside the bin directory.

Steps

1. Copy the Transformer cube to the production environment.

The name of the destination directory must be the same as the Transformer cube name. For example, if the cube is named cube_name.mdc, the destination directory must be named cube_name.

The location of the destination directory must be the same as where the cube is found when opened by IBM Cognos 8. For example, if the data source definition for the PowerCube in Content Manager indicates that the cube is found at d:\Cubes\Product.mdc, then the destination directory must be d:\Cubes\Product.

2. Open the Pcativate utility.

   The pcactivate.exe file is located in the *installation_location*\cognos\c8\bin folder.

3. At the command line prompt, type the following:

   pcactivate *cube_name.mdccube location*

   For example, type

   pcactivate product.mdc c:\cubes\product

Update Data Source Connection Information

You can update the data source connection information, with little to no impact on your IBM Cognos 8 users, using the IBM Cognos 8 PowerCube Connection Utility. When data source connection strings are changed, reports that reference the updated data source connection continue to run and end users begin seeing new cube data.

You can run the IBM Cognos 8 PowerCube Connection Utility to update PowerCube data source connections interactively, or as part of a batch script. Use the utility with cubes supported in IBM Cognos 8 version 8.3: IBM Cognos Series 7 and IBM Cognos 8 cubes, and on any platform supported by IBM Cognos 8 version 8.3.

Before you use the IBM Cognos 8 PowerCube Connection Utility to update published cube connections, ensure that your published PowerCube data source appears in the Administration tool.

Tip: When you use the IBM Cognos 8 PowerCube Connection Utility, you must press the Enter key at each prompt, even those prompts for which you enter values.

Steps to Manually Update a Data Source Connection

1. Open a command line window and go to the *installation_location*/c8/webapps/PCConn directory.

2. At the command line, type **PCConn** to launch the IBM Cognos 8 PowerCube Connection Utility.

   Tip: To view a list of available IBM Cognos 8 PowerCube Connection Utility commands, at the command prompt, type **help**.

3. To connect to a dispatcher, at the command prompt, type

   ```
   connect server:port
   ```

   where *server* and *port* represent the IBM Cognos 8 server and port number.

4. To connect to multiple dispatchers, repeat step 3 for each dispatcher to which you want to connect.

Tip: To view a list of dispatchers that you are currently connected to in the IBM Cognos 8 PowerCube Connection Utility, at the command prompt, type `connections`.

5. If your IBM Cognos 8 environment is secured and you are prompted for your logon credentials, enter the following authentication information:
   - namespace
   - user ID
   - password

6. If you want to review the current data source connection information, at the command prompt, type

   ```
   get data_source_name
   ```

   where *data_source_name* is the name of your PowerCube data source connection.

   The IBM Cognos 8 PowerCube Connection Utility returns the connection information for the data source, and allows you to see the connection string without opening the Administration tool.

7. At the command prompt, type

   ```
   set data_source_name
   ```

   You are prompted to type the new Windows or UNIX/Linux path.

   Tips:

   - To skip a Windows or UNIX/Linux path, press Enter.
   - To confirm the new data source connection information, repeat step 6.

8. Open IBM Cognos Connection, and go to the Administration tool.

9. Click the **Configuration** tab and click the data source.

10. Click the **Set properties** link.

11. On the **Connection** tab, review the new **Connection string** for the data source.

Overwrite a Published PowerCube

You can also use the IBM Cognos 8 PowerCube Connection Utility to enable, disable, or view the current state of a data source. To update the data source by overwriting it, disable it first. Use this approach when your process for updating published PowerCubes includes a staging folder. When a data source is disabled, end users cannot use any package that references the data source. When users attempt to use a package that references a disabled data source, they receive an error indicating that the data source cannot be found.

If a data source is in use when you attempt to disable it, you will receive a file lock error. To avoid this situation, we recommend that when you disable a data source, you also stop the Report and Report Batch services. Stopping the Report and Report Batch services affects not only the data source that is disabled, but all the data sources in the IBM Cognos 8 configuration. As a result, use this approach only during periods when your end users will not require access to the data sources.

Tip: When you use the IBM Cognos 8 PowerCube Connection Utility, you must press the Enter key at each prompt, even those prompts for which you enter values.

Steps to Manually Disable a Data Source for Overwriting

1. Open a command line window and go to the *installation_location*/c8/webapps/PCConn directory.
2. At the command line, type `PCConn` to launch the IBM Cognos 8 PowerCube Connection Utility.

   Tip: To view a list of available IBM Cognos 8 PowerCube Connection Utility commands, at the command prompt, type `help`, and press Enter.
3. To connect to a dispatcher, at the command prompt, type

   ```
   connect server:port
   ```

   where *server* and *port* represent the IBM Cognos 8 server and port number.
4. To connect to multiple dispatchers, repeat step 3 for each dispatcher to which you want to connect.

   Tip: To view a list of dispatchers that you are currently connected to in the IBM Cognos 8 PowerCube Connection Utility, at the command prompt, type **connections**.
5. If your IBM Cognos 8 environment is secured and you are prompted for your logon credentials, enter the following authentication information:
   - namespace
   - user ID
   - password
6. At the command prompt, type

   ```
   disable data_source_name
   ```

   Tip: To confirm that the PowerCube is disabled, at the command prompt, type

   ```
   state data_source_name.
   ```
7. To stop the Report and Report Batch services, at the command prompt, type

   ```
   stop reportService, batchReportService.
   ```
8. In your server environment, overwrite the PowerCube with the updated version.
9. To restart the Report and Batch Report services, at the command prompt, type

   ```
   start reportService, batchReportService.
   ```
10. At the command prompt, type

    ```
    enable data_source_name
    ```

Using Automated Processes to Update PowerCube Connection Information

You can automate the process of rebuilding and updating your published cubes with the IBM Cognos 8 PowerCube Connection Utility using batch processes and scripts. PowerCube update scripts can be combined with your cube build scripts, as well as with any ELT (extract, load, and transform) batch processes that you might have in place.

In large production environments, you may want to build PowerCubes on a server running Transformer that is separate from the server where the cubes are deployed. This allows you to optimize system resources for your cube builds and for your consumers, without the two environments competing for resources.

You can use commands in a text file to carry out the cube update process, and combine those commands with your Transformer cube building scripts to produce a fully automated cube build and update process.

The IBM Cognos 8 PowerCube Connection Utility takes commands from a text file using the `-f` command:

```
% PCConn -f commands.txt
```

Unless you add the -q command, the utility prints to stdout:

```
% PCConn -q -f commands.txt
```

Batch Process Workflow

Using batch processes to update published cubes with the IBM Cognos 8 PowerCube Connection Utility involves the following workflow:

❑ Build cubes on a separate server running Transformer using a staging folder.

❑ Rename the cube.

One option when renaming the cube is to append a date to the end of the cube name.

❑ Use FTP to transfer the cube to the IBM Cognos 8 server location referenced by the data source connection.

If you choose not to rename the cube as part of the batch process, consider deploying the cubes to folders named to allow for easy tracking.

❑ Use the set command to change the connection to point to the updated published PowerCube.

The IBM Cognos 8 Web studios will immediately recognize the new data source connection information without any further action by the modeler or administrator. Reports that reference the updated data source connection continue to run and end users begin seeing the new cube data, for example, during

- subsequent report runs
- active sessions during drill up/down or page up/down actions

Examples: Cube Update Script

Here are examples of a script using the set command. In these examples, the cube was already updated in Transformer.

Example - Allow Anonymous Access Set to True

```
connect <servername>:9300
logon -ns <namespace_ID> -uid <user_ID> -pwd <password>
set <data_source_name>
C:\cube_name1.mdc

exit
```

The empty return in the script indicates that there is no UNIX/Linux deployment path for the cube.

Example - Allow Anonymous Access Set to False

```
connect <servername>:9300
-ns <namespace_ID> -uid <user_ID> -pwd <password>
set <data_source_name>
C:\cube_name1.mdc

exit
```

The empty return in the script indicates that there is no UNIX/Linux deployment path for the cube.

Chapter 9: Guidelines for Optimizing Production Environments

This section provides optimization guidelines that have proven useful in enterprise-wide deployments. Organized as follows, these guidelines assume that your production system is implemented either on a UNIX or Linux platform, or in a mixed Windows/UNIX or Linux environment:

- Building PowerCubes in UNIX or Linux
- Controlling Processing with Preference Settings or Environment Variables
- Addressing Common UNIX and Linux Processing Problems
- Reducing Build Times for Large PowerCubes

When reviewing this material, be aware that run-time performance can vary, depending on the distance between your run-time environment, cubes, and data sources. It is also affected by the frequency and complexity of the OLAP reports that your users are running.

To achieve the fastest possible run-time performance, you may need to modify your system design. Consider enhancing your system by adding network bandwidth or computer power.

Network Bandwidth

Data flow between your data sources, cubes, and reporting computers can involve multiple networks and host computers. Average response times can be greatly affected by network capacity and the number of simultaneous requests to be handled. Daily fluctuations in your users' workload are just as important as the architecture used for your reporting system.

Computer Power

When doing capacity planning for the data servers and reporting computers used in your production environment, consider the volume of incoming OLAP queries and select hardware that will support your peak request loads. The cube structure is also important. Cubes with large partitions are only practical on servers that have the processing power to perform many calculations or queries per second.

Building PowerCubes in UNIX or Linux

Transformer (Windows) is the OLAP modeling component to use when designing and testing your prototype model, or when creating and reporting from cubes based on local data sources.

However, for large-scale production environments, you may want to move your OLAP reporting system to a more powerful environment, by specifying UNIX- or Linux-accessible data sources and then using Transformer on a UNIX or Linux platform to deploy and maintain your cubes on dedicated UNIX or Linux servers.

When running Transformer on UNIX or Linux rather than Windows, consider the following:

- Supported data sources include delimited and fixed-field text files, Cognos® PowerHouse® 4GL subfiles, IBM Cognos 8 packages and reports, IBM Cognos Impromptu Query Definition (.iqd) files (on IBM Cognos Series 7 version 7.4 supported platforms only) and other sources accessible through your database connectivity software. To coordinate cube builds with your data warehouse updates, we recommend that you run all related processes on UNIX/Linux servers.

- If you save your models as .mdl files, you can use an FTP program to transfer them from your Windows modeling computer to your UNIX or Linux production environment. You can then use the processing capability of your more powerful UNIX or Linux servers, to quickly and efficiently create cubes from one or more models, tuning your system to avoid network bottlenecks. You can run Transformer from the command line or by using Model Definition Language (MDL) scripts. Client-server modeling is not supported in Transformer version 8.x.

- If you need to add security to your cubes, you can leverage any authentication provider configured in IBM Cognos 8, apply custom views in Transformer for applicable users, roles, and groups, and deploy the resulting cubes to your UNIX or Linux environment. For more information about setting up and administering secure IBM Cognos 8 deployments, see the IBM Cognos 8 *Administration and Security Guide* and the *Architecture and Deployment Guide*.

Recommended Model Design, Cube Creation, and System Maintenance Process

❑ Design and create a prototype model locally, on a Windows computer.

❑ Create and test your cubes locally on a Windows computer.

❑ Define a UNIX- or Linux-accessible data source in the model and save your server cubes in Model Definition Language (MDL) format.

❑ Move your cubes to a UNIX or Linux production server and tune your system for optimal OLAP reporting performance.

Setting Up Data Sources for UNIX and Linux Cubes

Your models must be able to read data sources from the server where Transformer is running. Because Transformer is intended for production environments, it is likely that these data sources contain large volumes of data. When you build a prototype, you must set up source files that provide access to a subset of the data that is sufficient for designing your model.

The data sources you use to build the prototype must have the same columns as those you intend to use later, in the UNIX or Linux component. The names of columns used to define levels and measures in the models must match. However, the columns can be in any order and may be unused. If there are fewer categories in each column of the prototype source than in the production source, data records found in the production source which were not found in the prototype source will provide new categories for the UNIX or Linux server model.

IBM Cognos Impromptu Query Definition (.iqd) files and Framework Manager externalized queries (.iqd) files are available only on supported IBM Cognos Series 7 platforms. Specifically .iqd files are not a supported data source on Linux and HPUX Itanium platforms.

If you use data sources other than IQD files or IBM Cognos 8 package and report data sources, you must set up separate physical sources of flat file data sources on the local computer used for the prototype, and on your UNIX or Linux server. IBM Cognos 8 package and report data sources are maintained in the model; however, the UNIX or Linux server requires database client connectivity to retrieve query results for cube building.

Use IBM Cognos Reports to Create a Data Source

You can use Report Studio or Query Studio list reports, designed against DMR or relational data sources, as a data source in Transformer. When doing so, create the report to include prompts so that when testing the model, you can test a subset of the data that would be included in the full cube build.

Prompted reports are supported in Transformer, regardless of the platform on which the cube build is performed. Prompts from reports can be handled either interactively or silently as part of a batch script.

We recommend that you consider taking advantage of the data filtering capabilities of IBM Cognos 8 reports to

- filter by value on specific columns

  For example, you can include data for only a few of your regions when generating categories for the Regions dimension in the model.

- return a specific number of rows

  For example, if the server data source contains two million records, you can use only the first 10,000 records for your model prototype.

- return only unique records if the server data source contains a lot of duplicate records

  After you generate categories and create .mdl files and cubes for your production server, Transformer can acces the same database that you used to build the prototype model.

Example - Using an IBM Cognos 8 Data Source for a UNIX Cube

Suppose you have a sales analysis model with reports defined for Products, Regions, and Sales Transactions. You save these reports in IBM Cognos Connection, and use Transformer on Windows to read the data and define the model structure with a small number of text rows in Transformer.

You then use Transformer to rebuild the model using the report with no restrictions on data retrieval, build the cube, and deploy it to your UNIX server.

Steps

1. In IBM Cognos 8 Report Studio or Query Studio, create a new list report against a DMR or relational package that returns sufficient data to build a prototype model, using a prompt to filter the data results.

2. In Transformer on Windows, design and build the prototype, using the report as a source for the data source query.

3. Fill in the prompt information to filter the data results for testing purposes.

After you have the cube designed the way you want it, you can remove the filter using the prompt, or set the filter to the level of data required.

4. Build the cube.

5. If your production server is a different machine than the one on which you built the cube, use your standard file transfer protocol (FTP) procedure to upload the cube to your UNIX production environment.

Adding Security to a UNIX or Linux Cube

You can secure your cubes against the users, groups, and roles in your configured IBM Cognos 8 namespace, and update these security objects and any associated views in the UNIX or Linux environment by using Model Definition Language (MDL) scripts. For more information, see the Transformer *Developer Guide* and the IBM Cognos 8 *Administration and Security Guide*.

Note: If you are using a IBM Cognos Series 7 Access Manager namespace with Transformer version 8.x, the Content Manager must be on an IBM Cognos Series 7 supported platform. For information about supported UNIX and Linux server configurations, see the IBM Cognos 8 *Architecture and Deployment Guide*.

Although you can use a delete script to remove an entire namespace, to selectively remove obsolete security objects, you must modify your model in Transformer on Windows. The same applies if the contents of your namespace change, because such changes will affect your custom views.

Controlling Processing with Preference Settings or Environment Variables

You can invoke Transformer from the command line to populate models, create cubes, and perform other actions. How and where these actions are performed is determined by information that you provide.

You can control specific operating characteristics of Transformer by creating user preference files or by setting up environment variables, or by overriding either of these from the command line.

The settings contain information such as default directories for various classes of files that Transformer uses or creates.

All preference file settings can be used as environment variables, and you can use multiple preference files.

Transformer searches for settings in the following sequence:

1. cogtr.xml in the IBM Cognos configuration directory

2. environment variables

3. the `-d` or `-f` command line option

4. **Note:** If the command line contains both `-d` and `-f`, Transformer uses the one that appears last.

Environment variables override settings in any preference file, and command line options override environment variable settings.

For example, to override the directory where source files are read, you can include the following UNIX environment variable definition in the *.sh file:

```
DataSourceDirectory=$HOME/data; export $HOME/data
```

To use the command line to override the setting for the directory where the source files are read, start Transformer using the following command:

```
cogtr -dDataSourceDirectory=$HOME/data
```

Preference Settings and Environment Variables

Consider the following:

- Preference file syntax does not allow a space between `-d` and its *preference_var* argument.
- In preference files, blank characters, tab characters, and lines beginning with the number sign (#) are ignored.
- In most cases, preference files that contain invalid options are ignored, and an error message is written to the log file. For example, this error-handling applies if the preference file entry for the ModelSaveDirectory is incorrect. However, if the entry for CubeSaveDirectory is incorrect, the variable is ignored but no log file entry is added; if the ChildRatioThreshold entry is outside its acceptable range, the invalid setting is changed at runtime, to either its maximum or minimum value depending on which value is closest to the incorrect setting.
- Each environment variable must be defined before it is used in a command, and must be preceded by a dollar character ($). Optionally, braces ({}) may be used to enclose the environment variable name.
- The environment variable must be alphanumeric (ASCII-text format), and may contain an underscore (_).
- Special environment variable characters, such as $, {, or }, may appear in a file name or directory string if they are preceded by the escape character (\). This backslash is automatically removed before the string is used. For example, a pair of backslash characters (\\) is replaced by one backslash.
- Variable substitution is not performed on the values of environment variables.

Preferences

The following are preference file entries you can set for Transformer.

Note: In the following lists of preferences, *path* refers to the directory and file name. For non-Windows platforms, the default is the "temp" directory in the *installation_location*/c8 directory. On Windows platforms, different defaults apply as described below.

Directory

The following setting specifies where Transformer creates the temporary file while you work on your model. The default path is the value of ModelSaveDirectory. The temporary file can be used to recover a suspended model at strategic checkpoints, should a severe error occur during cube creation. This file has the extension .qy?, where the ? is replaced by the character used in your version of Transformer.

```
<Preference Name="ModelWorkDirectory" Value="
path1;path2;..."/>
```

The following setting specifies where Transformer creates temporary work files when generating cubes. The default path is the "temp" directory in the *installation_location*/c8 directory. Transformer creates multiple files automatically, and concatenates them into one logical file, regardless of which drive they are in.

```
<Preference Name="DataWorkDirectory" Value="path1;path2;..."/>
```

By distributing the files across multiple drives, you can work around size limitations imposed by your operating system, and reduce disk contention. The location of the files is determined by the list of paths that you specify. The files are created in the order specified in the list of paths.

For data source files other than .iqd-format files, the following setting specifies where Transformer searches for files. The default path is the data subdirectory in the *installation_location*/c8 directory.

```
<Preference Name="DataSourceDirectory" Value="path1;path2;..."/>
```

The following setting specifies where Transformer saves cubes. On Windows, the default is the \Transformer\PowerCubes subdirectory under your home directory. On UNIX and Linux, the default is the temp subdirectory in the *installation_location*/c8 directory.

```
<Preference Name="CubeSaveDirectory" Value="path1;path2;..."/>
```

The following setting specifies where Transformer saves models. On Windows, the default is the \Transformer\Models subdirectory under your home directory. On UNIX and Linux, the default is the temp subdirectory in the *installation_location*/c8 directory. If you are running Transformer with an .mdl file and you have specified the `-s` option, a .py? file is created in this directory. (The ? in the extension .py? is replaced by the character used in your version of Transformer, such as .pyj for versions 8.3 and 8.4.)

```
<Preference Name="ModelSaveDirectory" Value="path"/>
```

File

The following setting specifies the threshold number of bytes at which Transformer splits its work files. The minimum is 1000000000 and the maximum is 1500000000. The default value is 1500000000.

WorkFileMaxSize

The following setting specifies a value that determines one of the following options:

- the creation of a single cube that is larger than 2 GB
- the threshold at which multiple files are created for a very large cube; for example, 30,000,000 if your cube is still less than 2 GB or 1,000,000 to create smaller cubes

MultiFileCubeThreshold

This multiple-file cube has one file with the .mdc extension and one or more files with the .mdp (multi-dimension partition) extension. The cube cannot be compressed. The default MultiFileCubeThreshold value is 0, which signifies that multiple file generation is disabled.

Log File

The following settings control where and how Transformer writes information to an ASCII-text log file.

The following setting specifies where Transformer creates the log file. On Windows, the default is the \Transformer\logs subdirectory under your home directory. On UNIX and Linux, the default is the temp subdirectory in the *installation_location*/c8 directory. If the logs subdirectory does not exist, then the default is the current working directory.

```
<Preference Name="LogFileDirectory" Value="path"/>
```

The following setting specifies a file name if you want messages written to a file, rather than displayed on the screen. The file name can include the full path. By default, this file has the same name as the model file, but with a .log extension.

```
<Preference Name="LogFileName" Value="path"/>
```

The following setting specifies that the log file is overwritten for each new model or cube. A value of TRUE appends the new log data to the existing log file.

```
<Preference Name="LogFileAppend" Value="FALSE"/>
```

The following setting specifies the types of messages that are written to the log file. Choose from the following severity levels:

```
<Preference Name="LogDetailLevel" Value="4"/>
```

- 0 - suppresses logging
- 1 - includes severe errors only, which must result in corrupted files or inconsistent data

  These can arise if limits are exceeded for CPU, disk, file, or transaction resources, or if the model, cubes, or temporary files have been corrupted.
- 2 - includes severe errors and errors that occur at the transaction level

  These cause the cube to be invalid, but leave it in a consistent, possibly incomplete, state.
- 3 - includes severe errors and warnings indicating a potential problem

  Warning messages do not impede processing.
- 4 - includes severe errors, errors, warnings, and informational messages (default)

You can use the log file to check the status of cube creation. The progress of a cube update is indicated by statements in the file, each containing the following fields:

- date and time (24-hour clock) at which the message was issued
- the message severity
- the message text

The text of each message includes header information, the ID of the object being processed (in hex notation), and messages about Transformer processes and timing.

Messages marked "Timing" are especially useful to analyze as a series of processing events. You can do this by importing the log file into a spreadsheet application as a tab-and-comma-delimited file. For more information, see"Recommendation - Analyze Processing Bottlenecks Using the Log File" (p. 235).

You can also use the `-r` command line option to control the types of messages generated.

This setting specifies, in elapsed minutes, how often messages are written to the log file. The default is -1. When the setting is -1, messages are not logged. If it is set to 0, messages are written to the log file whenever they are generated:

```
<Preference Name="LoggingFrequency" Value="-1"/>
```

Warning

These settings control whether Transformer issues warnings about potential incremental update problems and ratios between categories and their descendants.

This setting issues warnings when an event is going to take place that will make an incrementally updated cube invalid; for example, deleting a category. The default value of TRUE means that warnings are issued; FALSE disables warnings.

```
<Preference Name="IncUpdateWarnings" Value="TRUE"/>
```

This setting issues a warning if the number of child categories for any parent category exceeds the specified value. Valid values are 1 through 16384. The default is 35.

```
<Preference Name="ChildRatioThreshold" Value="35"/>
```

Output

This setting specifies how often Transformer creates checkpoints for recovery from severe errors during cube creation. This is defined as the number of records written to a cube before a new checkpoint is created. If your data sources are constructed from a database, this value shouldn't exceed the size of your database rollback journal. The default is 500000.

```
<Preference Name="MaxTransactionNum" Value="50000"/>
```

Data Source Attributes

These settings specify various physical attributes of the data source files.

This setting specifies the character used as a decimal point in a data source. The default is a period (.).

```
<Preference Name="DecimalPoint" Value="."/>
```

This setting specifies the field delimiter in a delimited-text data file. The default is a comma (,).

```
<Preference Name="DefaultSeparator" Value=","/>
```

This setting specifies the character used as a thousands separator in a data source. The default is a comma (,).

```
<Preference Name="ThousandSeparator" Value=","/>
```

This setting specifies the cut-off date that determines whether the two-digit year (YY) in a six-digit date is a 20th or 21st century date. Transformer interprets values below the cut-off as 21st century dates and values at or above the cut-off as 20th century dates. The default is 20.

```
<Preference Name="CenturyBreak" Value="20"/>
```

Because 00 to 19 are automatically treated as 21st century dates and 20 to 99 as 20th century dates, you only need to change the default if your data source includes dates from 1900 to 1919. For example, the setting CenturyBreak=18 means that the values 00 to 17 are interpreted as 2000 to 2017 and the values 18 to 99 are interpreted as 1918 to 1999.

Date Format

This setting controls the format in which the date is displayed:

```
<Preference Name="LunarFiscalLabeling" Value="TRUE"/>
```

It determines whether users will be able to view dates in a cube in lunar year format. The value TRUE indicates that the dates will be displayed in this format. A value of FALSE indicates that dates will be displayed in calendar year format.

Environment Variables

The following sections describe the environment variables you can set for Transformer on UNIX and Linux. Shared library variables, and related database (RDBMS) variables server communication settings are omitted because client-server synchronization is not supported in IBM Cognos 8.

Shared Library Variables

To run Transformer on UNIX or Linux, the loader requires that the library path variable specify the location of the shared libraries. The library path variable for each supported operating system is as follows.

Operating System	Environment Variable
Sun Solaris	LD_LIBRARY_PATH
IBM AIX	LIBPATH
HP-UX	SHLIB_PATH
Linux	SHLIB_PATH

Database Variables

If your data sources make use of one or more relational databases, you must install them before using Transformer. For more information, see the *Installation Guide* for your version of Transformer.

Addressing Common UNIX and Linux Processing Problems

This section describes some tools and techniques for addressing common processing problems in the UNIX/Linux environment.

Checking UNIX and Linux Job Status

When you use Transformer to create cubes, there are several options for checking the cube processing status:

- You can view the contents of the log file, which contains messages issued by Transformer.
- If you used a `crontab` file or scheduled the job with the UNIX/Linux at command, you can check your email for a completion message, and review the log file for any warnings or errors.

- You can check your `ModelSaveDirectory` location for a checkpoint file (.qy?). Because Transformer automatically deletes checkpoint files when processing ends successfully, a checkpoint file means that a suspended model exists. Check the log file for errors associated with the processing of that model.

- If you created a shell script that includes `cogtr` commands, you can check the exit status to detect operations that did not end successfully. An exit status value of 0 indicates successful completion. Any other value indicates an error.

By default, all messages generated by Transformer are directed to the standard output stream. You can direct them to a log file instead, and control the properties of that log file with UNIX/Linux equivalents of the preferences set in the cogtr.xml files. You can analyze log file error and warning messages to help you isolate problems encountered when Transformer attempted to read the data source or write data to the cube.

Scheduling Batch UNIX and Linux Production Jobs

If you want to automate cube creation, you can use the UNIX or Linux scheduling commands, the UNIX/Linux `at` command, or a `crontab` file, to schedule these jobs at convenient times, in batch mode.

For example, suppose you have a model that you use to create 20 individual cubes and a cube group consisting of 10 cubes. You can supply scheduling information to enable and disable the creation of specific cubes at specific times.

Performing Incremental Updates on UNIX and Linux

If you update cubes incrementally and a cube update fails, if you try to repeat the most recent incremental update, some records from the update may already have been written to the cube. Restarting the process causes Transformer to add these records to the cube twice, producing inaccuracies.

Example - Updating a Model Using an MDL Script

This example demonstrates how to use an MDL script to split the generation of categories and the creation of cubes into separate processes.

The following MDL file contains the verb statements that are required to generate categories for the model go_sales.py?, without creating the cubes for that model.

```
OpenPY "go_sales.py?" PopulateModel SavePY "go_sales.py?"
```

If these statements are saved in a file called Gen_nat.mdl, you can process them by running Transformer, as follows:

```
cogtr -mGen_nat.mdl
```

Example - Restarting a Failed Process from a Checkpoint File

As Transformer processes data on the server, it maintains a checkpoint file with the extension .qy?. When a cube build or update fails, Transformer can use this .qy? file to restart processing at the point of failure.

For example, suppose you are running a quarterly model update with new data. Transformer is unable to locate one of the source files for a data source in the model, and the model update fails. You use the checkpoint file to restart processing at the point of failure.

The following command will resume processing of the .py? model file. If an associated .qy? file is found, it is used to restart the process:

```
cogtr -i go_sales.py?
```

For more information, see "Command Line Options" (p. 237).

Example - Restarting a Failed Process from the Beginning

When Transformer builds cubes, some conditions may result in a failed process. For example, if Transformer has insufficient disk space for the model files, data temporary files, and cubes, an error occurs and processing stops.

In such cases, you can restart the process from the beginning, using the following command. It loads the saved model file go_sales.py?, ignoring the last interrupted cube creation process and any associated .qy? files:

```
cogtr -i go_sales.py?
```

For more information, see the Command Line Options section.

Reducing Build Times for Large PowerCubes

This section describes techniques you can use to reduce build times when working with cubes having more than 500,000 categories, and parent:child ratios that deviate significantly from the ideal.

Included are examples to illustrate the efficiency gains you can expect from optimization strategies such as hardware upgrades, system tuning, memory re-allocation, or model redesign.

The main test model used to confirm these strategies had the characteristics described in the following table.

Test Model Characteristic	Description or Value
Total number of categories	492,152
Number of non-measure dimensions	5
Number of measures	5: 1 regular, 2 calculated, and 2 after-rollup calculated
Source data format	ASCII (tilde-character delimited)
Number of source files	9: 6 structural and 3 transactional; multiprocessing enabled for 4 largest
Number of transaction input records	50 million

Test Model Characteristic	Description or Value
Size of all source files	2.28 GB
Enabled options	crosstab-caching and auto-partitioning (5 passes with 500,000-record size limit)

The following grid shows the model design, as seen on the Transformer (Windows) interface:

Transportation (7) (2)	Date (197) (2)	HR Code (491,376) (2)	Country of Origin (286) (2)	Country of Export (286) (2)
Transportation (5 categories)	Year (15 categories)	HR Code 1 (1471 categories)	Region (10 categories)	Region (10 categories)
	Month (180 categories)	HR Code 2 (19036 categories)	Country (274 categories)	Country (274 categories)
		HR Code 3 (75916 categories)		
		HR Code 4 (123856 categories)		
		HR Code 5 (271095 categories)		

Recommendation - Multiprocessing with Transformer Server

By upgrading to a multi-CPU server and running multiple instances of Transformer in parallel, you can significantly decrease the build time for your largest cubes. In a model that generates more than one cube, the overall creation time is reduced to the time taken for the slowest cube to build.

We recommendthe following strategies:

- Give each Transformer process its own CPU. Because each instance uses system resources independently, ensure that each one has sufficient memory, I/O bandwidth, and disk space.
- Provide each Transformer instance with its own configuration files.
- Do not share the locations of `DataWorkDirectory` and `ModelWorkDirectory` among multiple instances.
- Add an ampersand (`&`) to the end of the UNIX/Linux command line to begin your first process in background mode. For example, use

  ```
  cogtr -mmodel.mdl &
  ```

  When control returns to the command prompt, initiate a second `cogtr` command in parallel.

- To continue processing on UNIX or Linux after session log off, type the command

  ```
  nohup cogtr -mmodel.mdl
  ```

Recommendation - Adjust the Memory Allocation

It is important to provide sufficient physical memory, on a server dedicated to building cubes, to avoid excessive pagination. In addition to having sufficient memory to handle all running application requests, make sure the operating system disk cache on the server can grow as large as required during the cube build process.

Once you have optimized the memory settings for your server, we recommend that you track virtual and working memory use, to find any bottlenecks created during the processing of categories in your largest slowest cubes.

Typically, total addressable (virtual) memory usage climbs rapidly during category generation, and remains relatively constant during all cube build phases (read data, update metadata, and update cube). Working memory, or that portion of the physical memory that is used by Transformer, also rises quickly and remains high until the cube update phase, when it is freed for use by the operating system disk cache.

In the first phase, the more categories there are, the more memory is required. Although models differ, working memory usage is typically 500-1,500 bytes per category. Paging (swap file use) occurs if the space allocated to working memory is insufficient and cannot grow to the amount required to process the model.

Recommendation - Reconfigure the Hard Drive

You can achieve additional gains by reconfiguring the hard drive of your build server to optimize I/O processing.

We recommend that you allocate at least three physical disk drives to your Transformer system, subdividing the processing as follows:

- Drive 1: operating system and the cogtr program
- Drive 2: Transformer `DataWorkDirectory`
- Drive 3: Transformer `ModelWorkDirectory` and `CubeSaveDirectory`

The log file for this configuration reads as follows, where 1, 2, and 3 represent drives c, d, and e:

```
LogFileDirectory=c:\transformer\logs\
ModelSaveDirectory=c:\transformer\models\
DataSourceDirectory=c:\transformer\data\
CubeSaveDirectory=e:\transformer\cubes\
DataWorkDirectory=d:\temp\
ModelWorkDirectory=e:\temp\
```

If you do not specify processing directories, Transformer version 8.x defaults to the IBM Cognos 8 default directories.

You may also want to configure the build server to use striping (RAID level 0) rather than mirroring, assuming that you have a backup for use if the system crashes during a production build. The build time is reduced because data is distributed among nonredundant disk drives.

Recommendation - Allocate Sufficient Space for the Temporary Files

You need different amounts of space for the temporary files during each cube build phase:

1. During the **Data Read** phase, the source data is read into a temporary work file based on the model structure. Insufficient disk space and database connectivity can cause problems at this stage.

2. During the **Metadata Update** phase, the contents of the temporary work file are compared to the categories in the model, to see which ones go into the cube, and a copy of the file is created. The original work file is only deleted after processing is complete and eligible categories are inserted into the cube. Insufficient disk space and lack of memory can cause problems at this stage.

3. During the **Data Update** phase, before the data points in the temporary work file can be inserted into the partitioned cube, the data must be sorted and consolidated. This requires several passes through the temporary file. The most common issue during this phase is low system memory.

There is a formula you can use to estimate the amount of space to allocate for the temporary files. If {([(A*4)+(B*4)+(C*16)+8]*D)/1024/1024}*2 is greater than 1431, the space calculation is that number plus E. If the calculated result is less than or equal to 1431, the required space is that number (do not add E).

Taking the test model as an example, the calculated space requirement is 7047 MB, given the following input values:

- A = the total number of dimensions
- B = the number of dimension views associated with the PowerCube
- C = the number of regular measures (calculated measures are not counted)
- D = the number of input records for all transactional data sources
- E = 1431; the `WorkFileMaxSize` setting divided by 1024 * 1024 (MB)

To this number, you must add sufficient space for the PowerCube and the model checkpoint files: 7 GB + 20% for the former, and a number equal to the model working space for the latter.

We recommend that you point the **Sort** directory to the same location as the **Data Temporary File**, with all other Transformer directory locations pointing to another drive. You should then use a system performance monitor, during the cube build, to check the amount of available disk space in the three cube build phases.

Recommendation - Optimize the Operating Environment

You can adjust various settings, whether operating in UNIX/Linux or Windows environments, to shorten your cube build times.

We recommend that you optimize your system by changing various default settings, as follows:

- WriteCacheSize

On UNIX or Linux servers, the default value for this setting is 32768 (32 MB). Doubling the value to 65536 (64 MB) or tripling it to 98304 (96 MB) is recommended to optimize larger UNIX or Linux systems.

To modify the `WriteCacheSize` setting, open the ppds_cfg.xml.sample file located in the *installation_location*/configuration directory. After you change the `WriteCacheSize` setting, save the file in the same directory as **ppds_cfg.xml**.

- Sort buffer size

  The sort buffer size used for local processing is specified as a preference setting on Windows and UNIX/Linux:

  - On Windows, set the **Work file sort buffer size** on the **File** tab of the **Preferences** dialog box.
  - On UNIX/Linux, set the sort buffer size using the `WorkFileSortSize` command.

  The default setting is 8000000; however, you can raise the amount of physical memory available for sorting data during the consolidation and auto-partitioning process.

- TEMPFILEDIRS (Windows only)

  You can change the location of the temporary files that are created whenever sorting is done. We recommend that you specify multiple directories, separating each with a semicolon.

- MaxTransactionNum (Windows only)

  This setting limits the number of records that can be processed in temporary files before a checkpoint is inserted and records are committed to the PowerCube. To reduce cube build time, try raising the setting from its default (500000) to 800000. Or, if you get error message TR0112 during a cube build, lower the setting so records are committed more frequently, thereby freeing up space.

  You change this setting on the **General** tab of the **Preferences** property sheet.

- Ulimit (UNIX/Linux only)

  Typically, you specify a value for this setting, such as 67 MB for a 2 GB-capacity server, so that system resources get shared effectively among competing processes. However, when operating Transformer on HP-UX, we recommend that you set this environment variable to unlimited, to provide the cube build process with as much physical memory as possible. (For other UNIX platforms or Linux, consult your operating system documentation to learn how you can best tune your kernel settings to allocate sufficient memory for Transformer.)

  Tip: Type the command `ulimit -a` to determine the currently assigned values for Transformer. The time, file, and memory settings should show a value of unlimited.

Recommendation - Redistribute Files

You can globally change various file location settings by means of a preference file, to place your files on servers with sufficient space to handle your production requirements.

You can also specify preference settings on the command line. These settings override or take precedence over all other settings, including environment settings defined in the cogtr.sh file, or the environment variables **TMPDIR**, **TEMP**, and **TMP** as defined by the operating system. If multiple variables are defined, Transformer uses the first one in the list.

The UNIX/Linux command line syntax for specifying global preferences is

```
cogtr -fpreferences.rc -mmodel.mdl
```

The Windows equivalent is

```
cogtr.exe -n -fc:\preferences.prf model.mdl
```

Recommendation - Optimize Gateway Settings for IBM Cognos Series 7 IQDs

To further shorten the data read phase for IBM Cognos Series 7 IQDs, you can change the database-specific settings found in the gateway .ini files included in the Transformer installation directory.

Search for a file name such as cogdm*.ini, where the asterisk represents a specific database version. The entries in each gateway .ini file are different, depending on the database type.

Note: For IBM Cognos 8 data sources, see the IBM Cognos 8 *Architecture and Deployment Guide*.

Example - Change Oracle Database Settings

Oracle uses the cogdmor.ini gateway file for database-specific settings. We recommend you consider adjusting the following settings:

- `Fetch Number of Rows`

  Increasing the number of rows to fetch in each fetch operation can improve performance on some systems. Although the current limit for this number is 32767, numbers larger than the default (100) may degrade performance on some systems.

- `Fetch Buffer Size`

  Increasing the size of buffer used during fetch operations from the default (2048 bytes) can improve performance on some systems.

Where both entries have been changed, the row setting takes precedence over the buffer size setting.

Recommendation - Keep Model and Cube Sizes Within Practical Limits

There are practical limitations on the file size of production models and cubes, based on typical memory capacities and run-time performance requirements, in the production environment.

For example, we recommend that you limit the size of your models to 2 million categories each. Pay particular attention to bloating due to excessive use of labels, descriptions, short names, category codes, and custom views. Metadata associated with your structural data source dimensions, levels, and categories can contribute significantly to overall storage requirements.

In virtual memory terms, models held in allocated memory are limited by the available address space. Also, performance is severely impacted if a model is larger than the available physical memory.

Recommendation - Analyze Processing Bottlenecks Using the Log File

The Transformer log file provides useful information to help you diagnose the cause of processing bottlenecks during cube builds.

We recommend that you complete the following procedure:

1. Import these delimited-field data log files into a spreadsheet, such as Microsoft Excel:

   LogFileDirectory=f: \logfiles\

2. In the resulting log file import, select the header of the E column and, if you are using Microsoft Excel, from the **Data** menu, click **Filter** and select the **AutoFilter** option.

3. In the list, select either (**NonBlanks**) or one of the three phases in the process. The spreadsheet now shows only the timing information for your selection.

4. Select several cells in the F column and select the **Sum** command, to add the timing values for your selected cells, as displayed in the lower toolbar.

5. Repeat the above steps to examine each subphase. These appear in distinct blocks as follows, with each completing before the next subphase begins:

   - **Data Read**

     INITIALIZING CATEGORIES

     OPEN DATA SOURCE

     READ DATA SOURCE

     MARKING CATEGORIES USED

   - **Metadata Update**

     SORTING

     UPDATE CATEGORY AND PROCESS WORK FILE

     METADATA

   - **Data Update**

     CUBE UPDATE

     CUBE COMMIT

   Note: If the timing shown for TOTAL TIME (CREATE CUBE) is different from the timing for the separate phases, you can adjust for the time difference in the Cube Update phase.

6. Assess how much time is spent on each phase. If you notice that, over time, cube builds are taking longer to complete even though the data set is relatively constant, compare each phase in the log files, to see where increases in build times are occurring.

Appendix A: Command Line Options

Transformer can perform certain modeling and cube-building tasks from the Windows, UNIX or Linux command line.

Note: You must invoke the command line utilities from the directory where the Transformer executable resides. In Transformer version 8.x, this is the bin directory.

This document provides the syntax for the following routine tasks:

- creating or updating cubes
- updating a model with new categories created during the category generation process
- running a set of batch jobs with different preference settings or input files
- changing the current date setting so that relative time calculations in batch cube creation are based on a specific date
- supplying database signon information
- changing the degree of detail for log file messages
- opening and deleting checkpoint files
- verifying and, if necessary, updating the scales used in MDL model columns and measures to match those used in the data source
- publishing cubes in Content Manager
- copying and activating new versions of published cubes
- supporting IBM Cognos 8 prompts
- opening specified .mdl files and executing MDL statements
- specifying the number of records for a test cube
- regenerating categories, and the measure scales used with them, without building the cube

Windows-only command line options enable you to

- run **AutoDesign**, build a model, create a cube, and display the results in a supported reporting component
- open the Transformer executable without showing a splash screen
- open the Transformer executable with the Main windows minimized

In IBM® Cognos® 8 Business Intelligence, cogtr replaces the IBM® Cognos® Series 7 Transformer executables (trnsfrmr.exe on Windows and rsserver on UNIX). However, the syntax is the same.

You can combine options using the space character to separate them. However, do not add spaces between an option and its argument, and enclose any intentional spaces found in an argument within double quotes. For example, the following is a valid Windows combination:

```
cogtr -n -k"field three=CarlosR/pw462" Field3.mdl
```

The following example, run from the bin folder where Transformer is installed, tests to ensure that your Windows operation ended successfully. If a non-zero error code is returned, the process failed, or Transformer terminated unexpectedly without completing the specified sequence.

```
cogtr.exe -n2 -s -i -nologo install_dir:\
filename.py? if errorlevel 1 goto error
:noerror echo no error was encountered pause goto end :error echo
Error
pause :end
```

In the following UNIX/Linux example, a Bourne Shell script is appended to the cogtr command to perform a second action (b) on successful completion of the first action (a):

```
#!/bin/sh if cogtr command_line_options then #perform action a if exit
status is 0 else #perform action b fi
```

Command Line Syntax

To use command line options, you must start the Transformer executable, cogtr.exe, from the directory in which it is installed.

The syntax for using command line options with optional arguments is as follows:

```
cogtr -option[argument] filename.py?|filename.mdl
```

Notes

When using command line options, the following considerations apply:

- On Windows, you can use -n with some options to run Transformer in batch mode.
- The option in the command line always starts with a dash (-).
- Command line options are case-insensitive. Arguments are case-sensitive.
- If there are spaces inside any argument, you must enclose the argument in double quotation marks, for example

  ```
  cogtr -n -k"field three=CarlosR/pw462" Field3.mdl
  ```

- For .py? files, the question mark (?) is replaced by the character that is used in your version of Transformer.
- You can use more than one option in a command line. If an option that is used in a command line is incompatible with an option that appears earlier in the command line, the earlier option is ignored.
- The *filename.py?* and *filename.mdl* syntax applies to Windows only; use the -m option on other platforms to specify the model file name.
- When entering command line options, you must specify the name of the .mdl and .py? file at the end of the command line after all the other parameters, for example,

  ```
  cogtr -n -s -mmodel.mdl
  ```

```
cogtr -n -s -pmodel.py?
```

Command Line Options

Transformer supports the following Windows and UNIX/Linux command line options. Detailed explanations are provided in the subsections that follow.

Command line options are case-insensitive.

Option	Meaning
-a	Runs AutoDesign then creates a cube, opens PowerPlay (Windows), and displays the report. `cogtr -a data_source` **Restriction:** Use on Windows only, with `-d`, `-f`, `-r`, and `-nologo` options.
-c	Generates categories and creates cubes. `-c -pfilename.py?|-mfilename.mdl` **Restriction:** Use with `-i`, `-m`, or `-p`. On Windows, use with `-n`.
-d	Overrides the specified user preference setting. `-dpreference_var=setting -pfilename.py?|-mfilename.mdl`
-e	Updates the model structure but not the data. `-e -pfilename.py?|-mfilename.mdl` **Restriction:** Cannot be used with `-c`. On Windows, use with `-n`. On Windows, Unix, and Linux, use with `-o`.
-f	Specifies the user-defined preference file. Can be used to publish PowerCubes in batch mode and include prompts in an XML command file using the XML schema for preference files. `-fpreference_file -pfilename.py?|-mfilename.mdl`
-g	Copies new versions of cubes to deployment locations and activates the newer versions. `-g[powercube_name]|[powercube_group_name/child_cube_name] -pfilename.py?|-mfilename.mdl`
-i	Opens the specified .py? model and restarts a failed process from the beginning. `-i -pfilename.py?` **Restriction:** On Windows, use with `-n`. On UNIX and Linux, cannot be used with `-s`.

Option	Meaning
-h	Displays help for the command line. `cogtr -h`
-k	Specifies database signon information for IBM Cognos Series 7. `-ksignon=userid/password -pfilename.py?|-mfilename.mdl`
-l	Specifies user authentication information for IBM Cognos 8. `-lsignon=userid/password -pfilename.py?|-mfilename.mdl`
-m	Opens the specified .mdl file or accepts Model Definition Language (MDL) statements. `-mfilename.mdl`
-n	Runs cogtr in interactive or batch mode. `-ndisplay_state -pfilename.py?|-mfilename.mdl` **Restriction:** Use on Windows only.
-nologo	Omits splash screen display when opening Transformer. `-nologo -pfilename.py?|-mfilename.mdl` **Restriction:** Use on Windows only.
-o	Turns off various model and cube creation actions. `-o -pfilename.py?|-mfilename.mdl`
-ox	Disables category creation and cube rebuild. `-ox -pfilename.py?|-mfilename.mdl` **Restriction:** Use on Windows only.
-p	Opens the specified binary model file, .py?, where the question mark (?) is replaced by the character that is used in your version of Transformer `-pfilename.py?` **Restriction:** Not valid with an MDL file.
-r	Specifies the level of detail for error-logging reports. `-rlog_level -pfilename.py?|-mfilename.mdl` **Restriction:** Valid levels are 0, 1, 2, 3, and 4.
-s	Saves the model. `-s -pfilename.py?|-mfilename.mdl` **Restriction:** On Windows, use with `-n`; cannot be used with `-i` or `-p`.

Option	Meaning
-t	Sets the current period. `-tcategory_code -pfilename.py?|-mfilename.mdl` **Restriction:** On Windows, use with `-n`.
-u	Gets the partition status for previously generated cubes. `-u[powercube_name]|[powercube_group_name/child_cube_name] -pfilename.py?|-mfilename.mdl` **Restriction:** Cannot use this option with secured cubes. On Windows, use with `-n`.
-v	Specifies the number of records for the test cube. `-vdata_subset_number` **Restriction:** Use with `-c`, `-m`, or `-p`. On Windows, use with `-n`.
-x	Updates the column and measure scales based on the data source. `-x -mfilename.mdl`

-y	Specifies how IBM Cognos Series 7 user-class security conversion is performed.
	● Use `-y1` to preserve both the IBM Cognos Series 7 user classes and custom views associated with the IBM Cognos Series 7 model.
	`-y1namespaceName=username/password`
	● Use `-y2` to preserve only the custom view associated with the IBM Cognos Series 7 model.
	`cogtr.exe -n2 -ox -s -y2 -mfilename.mdl`
	● Use `-y3` to discard the IBM Cognos Series 7 user classes and custom views associated with the IBM Cognos Series 7 model.
	`cogtr.exe -n2 -ox -s -y3 -mfilename.mdl`

-a option

This Windows-only option runs the **AutoDesign** utility, builds a model based on the specified data source, and creates the PowerCube.

Use this option with the `-d`, `-f`, `-r`, and `-nologo` options. Leave a space between the `-a` and the data source file name.

The syntax for using the `-a` option is as follows:

```
cogtr -a data_source
```

The following Windows example opens the Sales data source and uses **AutoDesign** to place dates in the time dimension, columns with numeric values in the **Measures** list, and all remaining columns in the dimension map, using a best-fit approach.

It then creates the **Sales** cube and if PowerPlay (Windows) is installed on the same computer, displays the Sales report in the reporting application.

```
cogtr -a Sales_datasource
```

-c option

This option loads a model file, interprets MDL statements, generates categories, and creates cubes.

Use this option with the applicable file-opening option: either -p, -m, or -i. On Windows, use this option with -n.

The syntax for using the -c option is as follows:

```
cogtr -c -pfilename.py?|-m
filename.mdl
```

The following UNIX/Linux example uses -c and -p options together to open the binary model file go_sales.pyj and process it as described.

```
cogtr -c -pgo_sales.pyj
```

The following UNIX/Linux example uses the -c and -m options together to open the equivalent full model definition (the .mdl text file for go_sales), and process it as described.

```
cogtr -c -mgo_sales.mdl
```

-d option

This option sets a new value for a Transformer user preference. The value overrides settings from the **Preferences** property sheet for this instance only.

No space may appear between the -d option and its argument. The argument is case-sensitive and must match the value specified in the cogtr.xml file.

For example, use -dLogFileName not -dlogfilename. The syntax for using the -d option is as follows:

```
cogtr -dpreference_var=
setting -pfilename.py?|-m
filename.mdl
```

If you specify the -d option after the -f option, the -d setting overrides the setting for -f. The reverse is also true; the last-appearing option overrides the options that precede it.

You can set preferences in the command line, environment variables, and cogtr.xml. The priority of settings is that the setting in the command line takes precedence over the setting in the environment variable, and the setting in the environment variable takes precedence over the setting in cogtr.xml. If you set a preference more than once in a command line, the last setting overwrites all previous settings.

You can use most settings in the cogtr.xml file as arguments for *preference_var*.

Although you can specify several -d arguments on the same command line, for Windows production jobs, we recommend that you create your own preference file based on the settings in the cogtr.xml file. You can then reference this file, using the -f option, to run batch Transformer jobs. Similarly, on UNIX and Linux, you can set global preferences using environment variables.

There is a sample preferences file, cogtr.xml.sample in the *installation_location*/configuration directory. The actual preferences file, cogtr.xml, is not installed by default. It is created and saved to the *installation_location*/configuration directory the first time you save changes to the **Preferences** property sheet in Windows. For more information, see "cogtr.xml File Settings" (p. 347).

The following example overrides the preference file setting for the DataSourceDirectory, changing it to C:\Newdata. On Windows, it opens the Transact.mdl model and creates the cubes defined there, in batch mode.

```
cogtr -dDataSourceDirectory=C:\Newdata -nTransact.mdl
```

The following example overrides the default value at which a warning is issued, for a parent category having too many descendants. The new preference setting (threshold) is 25 children.

```
cogtr -dChildRatioThreshold=25 -nTransact.mdl
```

Note: If you change a preference using the `-d` option, you cannot use the **Preferences** property sheet in Windows to edit the preference that you changed. The Windows control for the preference that you changed using the `-d` option is inactive.

Preference Settings or Environment Variables

You can also use environment variables to set preferences. Transformer version 8.x recognizes the following preference settings or environment variables. For more information, see "Controlling Processing with Preference Settings or Environment Variables" (p. 222).

Preference or Environment Variable	Settings
CenturyBreak	**Default:** 20
ChildRatioThreshold	**Default:** 35 **Minimum:** 1 **Maximum:** 4294967295
CubeSaveDirectory	**Default:** On Windows, the default is the \Transformer\PowerCubes subdirectory under your home directory. On UNIX and Linux, the default is the temp subdirectory in the *installation_location*/c8 directory.
DataSourceDirectory	**Default:** the data subdirectory in the *installation_location*/c8 directory
DataWorkDirectory	**Default:** the temp subdirectory in the *installation_location*/c8 directory
DecimalPoint	**Default:** a period (.)
DefaultSeparator	**Default:** a comma (,)

Preference or Environment Variable	Settings
FilenameVariables	**Default:** FALSE
IncUpdateWarning	**Default:** TRUE
LogDetailLevel	**Default:** 4 **Minimum:** 0 **Maximum:** 4
LogFileAppend	**Default:** FALSE
LogFileDirectory	**Default:** On Windows, the default is the \Transformer\logs subdirectory under your home directory. On UNIX and Linux, the default is the logs subdirectory in the *installation_location*/c8 directory. If the logs subdirectory does not exist, then the default is the current working directory.
LogFileName	**Default:** " "
LoggingFrequency	**Default:** -1
LunarFiscalLabeling	**Default:** TRUE
MaxTransactionNum	**Default:** 50000
ModelSaveDirectory	**Default:** On Windows, the default is the \Transformer\Models subdirectory under your home directory. On UNIX and Linux, the default is the temp subdirectory in the *installation_location*/c8 directory.
ModelWorkDirectory	**Default:** the temp subdirectory in the *installation_location*/c8 directory
MultiFileCubeThreshold	**Default:** 0 (disabled) **Minimum:** 0 **Maximum:** 4294967295
OrderByCategoryLabeling	**Default:** 0 (disabled)
PPDS_READ_MEMORY	**Default:** 8000000 **Minimum:** 1000000

Preference or Environment Variable	Settings
PPDS_WRITE_MEMORY	**Default:** 4000000 **Minimum:** 1000000
ThousandSeparator	**Default:** a comma (,)
WorkFileMaxSize	**Default:** 1500000000 **Minimum:** 100000000 **Maximum:** 1500000000

-e option

This option updates and saves all the cube metadata that is defined in the model, but does not update the data. The cube metadata consists of object names, labels, short names, descriptions, drill-through reports, and security information.

You cannot use this option with `-c`, and you should always use it in combination with the `-n` and `-o` options, to run in Windows batch mode and suppress cube creation. Otherwise, you defeat the purpose of this command.

The syntax for using the `-e` option is as follows:

```
cogtr -e -pfilename.py?|-m
filename.mdl
```

The following Windows example opens the New.pyj model file and updates the defined cubes without regenerating existing categories or creating new ones. It then saves the model file along with its updated cube metadata: that is, the object names, labels, descriptions, drill-through reports, and security information.

```
cogtr -e -n -o -pNew.pyj
```

-f option

This option specifies the user-defined preference file or files to use. If you do not include the full directory path with the file name, Transformer searches the executable directory of your most recently installed rendition of the product for the required .xml file.

This option is also used to publish PowerCubes in batch mode and include prompts in an XML command file using the XML schema for preference files.

If you are performing batch tasks that require the use of multiple preference files, Transformer combines the settings in each file successively; later settings override previously defined ones.

Similarly, if you specify the `-f` option after the `-d` option, the `-f` setting overrides the setting for `-d`. The reverse is also true; the last-appearing option overrides the options that precede it.

No space may appear between the `-f` option and its argument, *`preference_file`*.

Tip: You can base your preference file entries on the settings the cogtr.xml file, and then run batch jobs by referencing the appropriate file using the `-f` option. Set your environment variables globally or use the `-d` option for specific command sequences.

The syntax for using the `-f` option is as follows:

```
cogtr -fpreference_file -p
filename.py?|-mfilename.mdl
```

The following example sets the preference file to C:\Monthly.xml, opens the model file Transact.mdl, then runs the process in batch to create all of the cubes defined in the model:

```
cogtr -n -fc:\Monthly.xml -mTransact.mdl
```

The following UNIX/Linux example parses the go_sales.mdl file using mypref.prf as the specified preference file:

```
cogtr -fmypref.prf -mgo_sales.mdl
```

Note: If you change a preference using the `-f` option, you cannot use the **Preferences** property sheet in Windows to edit the preference that you changed. The Windows control for the preference that you changed using the `-f` option is inactive.

XML Schema for Preference Files

The XML file format supports multi-value commands and user-defined preference files. Commands in XML files are executed sequentially unless specific rules are defined.

The XML schema can have multiple sections and multiple commands. Commands can contain multiple parameters and parameters can have multiple values.

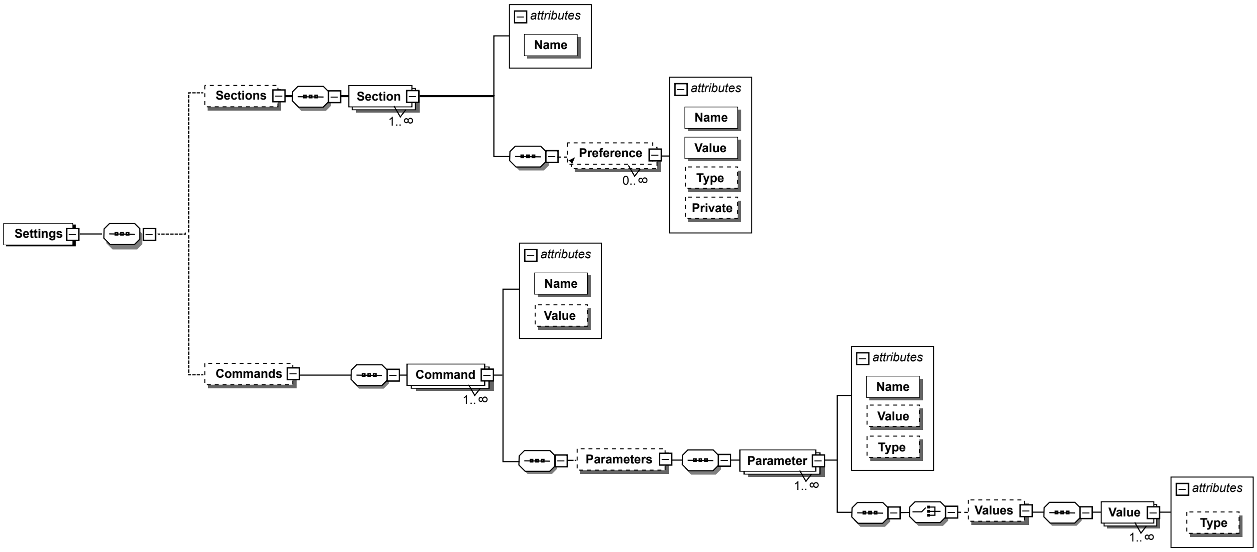
The cogtr.xml file conforms to the XML schema. The cogtr.xml file contains two major predefined groups of xml elements:

- a section which contains a list of preferences, for example:

```
<Sections>
<Section Name="Transformer">
<Preference Name="DataWorkDirectory" Value="..\temp"/>
<Preference Name="AutoEdit"Value="0"/>
<Preference Name="ChildRatioThreshold" Value="35"/>
<Preference Name="CubeSaveDirectory" Value="..\temp"/>
<Preference Name="DataSourceDirectory" Value="..\temp"/>
</Section><Section Name="RecentFileList">
<Preference Name="File1" Value="NationalOriginal.mdl"/>
<Preference Name="File2" Value="Cubexx.mdl"/>
<Preference Name="File3" Value="GreatOutdoorsCompany_Slow_v1.mdl"/>
<Preference Name="File4" Value="testcube.mdl"/>
</Section>
</Sections>
```

- a Commands section, which will be empty in most cases. The Commands section passes commands to Transformer when it is used in batch mode.

The following diagram shows the XML schema.



Example

The following example shows multiple preferences and commands being passed to Transformer.

```
<?xml version="1.0" encoding="UTF-8"?>
<!--Sample XML file generated by XMLSpy v2007 sp2 (http://www.altova.com)-->
<Settings xsi:noNamespaceSchemaLocation="cogtr_format_v2.xsd" xmlns:xsi="http:
//www.w3.org/2001/XMLSchema-instance">
 <Sections>
  <Section Name="Transformer">
   <Preference Name="DataWorkDirectory" Value="..\temp"/>
   <Preference Name="AutoEdit" Value="0"/>
   <Preference Name="ChildRatioThreshold" Value="35"/>
   <Preference Name="CubeSaveDirectory" Value="..\temp"/>
   <Preference Name="DataSourceDirectory" Value="..\temp"/>
  </Section>
  <Section Name="RecentPackageList">
   <Preference Name="mru_entry_0" Value="/content/package[@name=&apos;
EquifaxCube&apos;]"/>
  </Section>
  <Section Name="RecentFileList">
   <Preference Name="File1" Value="c:\NationalOriginal.mdl"/>
   <Preference Name="File2" Value="c:\Modified Cubexx.mdl"/>
   <Preference Name="File3" Value="c:\GreatOutdoorsCompany_Slow_v1.mdl"/>
   <Preference Name="File4" Value="c:\testcube.mdl"/>
  </Section>
 </Sections>
 <Commands>
  <Command Name="Publish">
   <Parameters>
    <Parameter Name="CubeName" Value="test"/>
    <Parameter Name="CognosConnectionDataSourceName" Value="test"/>
    <Parameter Name="DataSourceWindowsLocation"
     Value="c:\test1.mdc"/>
    <Parameter Name="DataSourceUnixLinuxLocation" Value=""/>
    <Parameter Name="DataSourceNameSpace" Value=""/>
    <Parameter Name="ReadCacheSize" Value="0"/>
    <Parameter Name="DataSourceSignon" Value="FALSE"/>
    <Parameter Name="DataSourceDescription" Value=""/>
    <Parameter Name="DataSourceToolTip" Value=""/>
    <Parameter Name="DataSourceUpdate" Value="FALSE"/>
    <Parameter Name="PackageName" Value="tpc"/>
    <Parameter Name="Packagelocation" Value="/content"/>
    <Parameter Name="PackageDescription" Value=""/>
    <Parameter Name="PackageToolTip" Value=""/>
    <Parameter Name="PackageUpdate" Value="FALSE"/>
    <Parameter Name="PackageAllowNullSuppression" Value="TRUE"/>
    <Parameter Name="PackageAllowMultiEdgeSuppression"
     Value="TRUE"/>
    <Parameter Name="PackageAllowAccessToSuppressionOptions"
     Value="TRUE"/>
   </Parameters>
  </Command>
  <Command Name="n"/>
  <Command Name="d" Value="DataSourceDirectory=C:\Newdata"/>
  <Command Name="prompts" Value="MyQuery1">
   <Parameters>
    <Parameter Name="Prompt1" Value="TestCube"/>
    <Parameter Name="Prompt2">
     <Values>
      <Value>c:\test\cube.mdc</Value>
      <Value>c:\test\cube1.mdc</Value>
      <Value>c:\test\cube2.mdc</Value>
     </Values>
    </Parameter>
    <Parameter Name="Prompt3" Value="TestPackage"/>
   </Parameters>
  </Command>
  <Command Name="prompts" Value="MyQuery2">
   <Parameters>
    <Parameter Name="Prompt4">
     <Values>
      <Value>1</Value>
      <Value>4</Value>
      <Value>8</Value>
     </Values>
    </Parameter>
   </Parameters>
  </Command>
 </Commands>
</Settings>
```

Publishing in Batch-mode

This command allows users to publish PowerCubes in batch mode.

The syntax is

```
cogtr -fcommand file name -p
filename.py?|-mfilename.mdl
```

where *command file name* is the name of the file containing the Publish command specification.

The following table describes the command parameters.

Parameter name	Description
CubeName	The name of the PowerCube in the model. This is a mandatory value.

Parameter name	Description
`CognosConnectionDataSourceName`	The name of the Content Management data source. This is a mandatory value. The set will fail if it is not defined. **Default:** The name in the model
`DataSourceWindowsLocation`	The Windows location of the data source. **Default:** The current cube location on Windows
`DataSourceUnixLinuxLocation`	The UNIX/Linux location of the data source. **Default:** Empty
`ReadCacheSize`	The read cache size for PPDS. **Default:** 0
`DataSourceNameSpace`	The authentication namespace for the Content Management data source. **Default:** Empty
`DataSourceSignon`	The command used to create the Content Management data source signon, if needed. **Default:** False
`DataSourceDescription`	The description of the Content Management data source. **Default:** Empty
`DataSourceToolTip`	The tool tip for the Content Management data source. **Default:** Empty
`DataSourceUpdate`	The command used to update a data source. **Default:** FALSE
`PackageName`	The name of the Content Management package. This is a mandatory value. The set will fail if it is not defined. **Default:** The name in the model
`Packagelocation`	The location of the Content Management package. **Default:** Public Folders

Parameter name	Description
PackageDescription	The description of the Content Management package. **Default:** Empty
PackageToolTip	The tool tip for the Content Management package. **Default:** Empty
PackageUpdate	The command used to update the existing package when its settings change. **Default:** FALSE
PackageAllowNullSuppression	The command used to specify whether suppression is available to IBM Cognos 8 studio users when working with this package. **Default:** TRUE
PackageAllowMultiEdgeSuppression	The command used to specify whether IBM Cognos 8 studio users can suppress both rows and columns. If this option is set to FALSE, users can suppress rows only or columns only. If this parameter is set to TRUE, the PackageAllowNullSuppression parameter must also be set to TRUE. **Default:** TRUE
PackageAllowAccessToSuppressionOptions	The command used to specify whether IBM Cognos 8 studio users can control the types of empty values that will be suppressed, such as zero or missing values. Types of empty values that users can choose to suppress depend on the IBM Cognos 8 studio. If this parameter is set to TRUE, the PackageAllowNullSuppression parameter must also be set to TRUE. **Default:** TRUE

Example

The following example shows the xml code for publishing a cube.

```
<Command Name="Publish">
 <Parameter Name="CubeName" Value="NATIONAL"/>
 <Parameter Name="CognosConnectionDataSourceName" Value="NATIONAL"/>
 <Parameter Name="DataSourceWindowsLocation" Value="c:\test\cube.mdc"/>
 <Parameter Name="DataSourceUnixLinuxLocation" Value=""/>
 <Parameter Name="DataSourceNameSpace" Value=""/>
 <Parameter Name="ReadCacheSize" Value="0"/>
 <Parameter Name="DataSourceSignon" Value="FALSE"/>
 <Parameter Name="DataSourceDescription" Value=""/>
 <Parameter Name="DataSourceToolTip" Value=""/>
 <Parameter Name="DataSourceUpdate" Value="FALSE"/>
 <Parameter Name="PackageName" Value="NATIONAL"/>
 <Parameter Name="Packagelocation" Value=""/>
 <Parameter Name="PackageDescription" Value=""/>
 <Parameter Name="PackageToolTip" Value=""/>
 <Parameter Name="PackageUpdate Value="FALSE"/>
 <Parameter Name="PackageAllowNullSuppression" Value="TRUE"/>
 <Parameter Name="PackageAllowMultiEdgeSuppression" Value="TRUE"/>
 <Parameter Name="PackageAllowAccessToSuppressionOptions" Value="TRUE"/>
</Command>
```

Including Transformer Version 8.x Prompts in an XML Command File

You can include prompts in an XML command file. You must use the following command line in order for Transformer to read the file:

```
cogtr -fcommand file name
```

where *command file name* contains a sequence of statements that define prompt values.

The command file can contain one or more commands for prompts. The command name is `prompt`. The value attribute of the command specifies the prompt name. Each `prompt` command contains one or more `Parameter` elements that specify a query name, prompt attributes and values. The Query `Parameter` element specifies the query to which the prompt belongs. The other `Parameter` elements define the prompt type and values.

There are several different types of prompts: simple, multi-valued, range, and MUN.

The following example shows single value, multi-value and range prompts.

Prompts can be included in an XML command file. You must use the following command line in order for Transformer to read the file:

```
cogtr -fcommand file name
```

where *command file name* contains a sequence of statements that define prompt values.

The Command File can contain one or more commands for prompts. The command name is `prompt`. The value attribute of the command specifies the prompt name. Each `prompt` command contains one or more `Parameter` elements that specify a query name, prompt attributes and values. The Query `Parameter` element specifies which query the prompt belongs to. The other `Parameter` elements define the prompt type and values.

There are several different types of prompts: simple, multi-valued, range, and MUN.

The following example shows single value, multi-value and range prompts.

```
<?xml version="1.0" encoding="UTF-8"?>
<Settings xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
 <Commands>
 <!-- SINGLE_VALUE PROMPT -->
  <!-- Prompt name -->
  <Command Name="prompt" Value="MyPrompt1">
   <Parameters>
    <!-- A query this prompt belongs to -->
    <Parameter Name="Query" Value="Promptmany~1"/>
    <!-- SingleValue, MultiValue, Range -->
    <Parameter Name="PromptType" Value="SingleValue"/>
    <!-- Any type that understood by RS. Optional. Not in use -->
    <Parameter Name="PromptValueType" Value ="Integer"/>
    <!-- Value -->
    <Parameter Name="PromptValue" Value="12345"/>
   </Parameters>
  </Command>
 <!-- MULTI_VALUE PROMPT -->
  <!-- Prompt name -->
  <Command Name="prompt" Value="MyPrompt2">
   <Parameters>
    <!-- A query this prompt belongs to -->
    <Parameter Name="Query" Value="Promptmany~1"/>
    <!-- SingleValue, MultiValue, Range -->
    <Paramete Name="PromptType" Value="MultiValue"/>
    <!-- Any type that understood by RS. Optional. Not in use -->
    <Parameter Name="PromptValueType" Value ="String"/>
    <Parameter Name="Values">
     <Values>
      <Value>abc</Value>
      <Value>bcd</Value>
      <Value>cdf</Value>
      <Value>dfe</Value>
      </Values>
    </Parameter>
   </Parameters>
  </Command>
 <!-- RANGE -->
  <!-- Promptname -->
  <Command Name="prompt" Value="RangePrompt">
   <Parameters>
    <!-- A query this prompt belongs to -->
    <Parameter Name="Query" Value="Promptmany~1"/>
    <!-- SingleValue, MultiValue, Range -->
    <Parameter Name="PromptType" Value="Range"/>
    <!-- Any type that understood by RS. Optional. Not in use -->
    <Parameter Name="PromptValueType" Value ="String"/>
    <Parameter Name="Values">
     <Values>
      <Value>Range - form</Value>
      <Value>Range - to</Value>
     </Values>
    </Parameter>
   </Parameters>
  </Command>
 </Commands>
</Settings>
```

Simple Prompt

A simple prompt is a variable that has a single value. For example

```
<Parameter Name="SimplePrompt" Value="Single Value"/>
```

Multi-valued Prompt

A multi-valued prompt is a variable that has a number of different values. For example

```
<Parameter Name="Multi-valued-variable">
 <Values>
   <Value>Value1</Value>
   <Value>Value2</Value>
   <Value>Value3</Value>
 </Values>
</Parameter>
```

Range Prompt

A range prompt is a type of multi-valued prompt. It contains a Range-from and a Range-to value. For example

```
<Parameter Name="Range-variable">
 <Values>
    <Value>Range-from</Value>
    <Value>Range-to</Value>
 </Values>
</Parameter>
```

A range prompt does not need to contain both Range-from and Range-to values, but it must have one or the other, as shown below.

```
<Parameter Name="Range-variable">
 <Values>
   <Value/>
   <Value>Range-to</Value>
 </Values>
</Parameter>
```

MUN Prompt

A MUN prompt is a type of single-value prompt. It must refer to a Member Unique Name, for example

```
<Parameter Name="[ASPV]" Value="[AMERICA].[WORLD]"/>
```

-g option

This option copies newer versions of cubes to deployment locations and activates the newer versions.

If you want to create the newer version of the cube and then copy and activate it, use this -g option with the `-c` option.

The syntax for using the `-g` option is as follows:

```
cogtr -g[powercube_name]|[
powercube_group_name/child_cube_name] -p
filename.py?|-mfilename.mdl
```

The *powercube_name* parameter represents the name of the cube in the Transformer model. If *powercube_name* specifies a cube group that is not a time-based partitioned cube, all child cubes of the cube group are copied to the deployment location and activated. If *powercube_name* specifies a time-based partitioned cube, it is processed as a single cube; member cubes are not deployed individually.

If a name conflict exists with a child cube of a cube group that has the same name as another cube in the model, use the *powercube_group_name/child_cube_name* parameters.

When you use the -g option without specifying a cube name, all cubes in the model that have deployment locations defined and enabled, and proper status, are copied and activated.

The following example copies and activates all cubes in Sales.pyj.

```
cogtr -g -pSales.pyj
```

The following example copies and activates the child cube named "France" in a cube group named "Europe".

```
cogtr -gEurope/France -pSales.pyj
```

-h option

This option is displays help for the command line.

On UNIX and Linux, help is also displayed if you do not provide any command options.

The syntax for using the `-h` option is as follows:

```
cogtr -h
```

-i option

This option opens a saved model, regardless of the existence of a checkpoint file. On UNIX and Linux, you cannot use the `-i` option with the `-s` option.

A checkpoint file is automatically created when a model is suspended due to a general system failure, such as that caused by a power outage. The next time the model is opened in interactive mode, users can open either the checkpoint file or the last-saved version of the model file. In batch mode, you can use the `-i` option to bypass the prompt and force Transformer to open the original model file instead of the checkpoint file.

The syntax for using the `-i` option is as follows:

```
cogtr -i filename.py?
```

Note: Checkpoint files have a .qy? extension. As with .py? files, the ? (question mark) in the extension is replaced by the character that is used in your release of Transformer, such as .qyj.

The following Windows example opens the model file, Sales.py?, discarding any existing checkpoint file, then runs the process in batch mode to create all defined cubes:

```
cogtr -n -i Sales.py?
```

-k option

This option supplies the signon information needed to establish one or more database connections during batch processing. It provides an alternative to storing the information in the **Signons** list or retrieving user IDs and passwords from the configured authentication source. This is especially useful when used with .mdl files which, by default, do not use verb MDL and so do not store signon passwords.

For databases referenced in an Impromptu Query Definition (IQD) file, the signon is the logical database name that appears in the related Transformer .xml file. Multiple IQD data sources can use the same signon object.

Note: You can view these signons on the Transformer (Windows) interface, but you cannot change them.

The syntax for using the `-k` option is as follows:

```
cogtr -ksignon=
userid/password -p
filename.py?|-mfilename.mdl
```

No spaces may appear between `-k` and its argument, and the signon name cannot be empty. Also, the signon name cannot contain the ASCII equals (=) character, and the user ID cannot contain a forward slash (/), because both characters are reserved for the parameter syntax.

The following Windows example opens the Field3.mdl file and grants access to the required database using the signon named `field`, user ID `CarlosR`, and password `pw462`. Assuming that the signon matches that defined in the model, the process runs in batch mode, creating the cubes as defined in the model.

```
cogtr -n -kfield=CarlosR/pw462 Field3.mdl
```

The following UNIX example reads the data source for model Xyzsales.mdl from an Oracle database using an IQD file, confirms the security information, and processes the model in batch mode. The signon `sal_log` includes the Oracle user ID `corpadm` and the password `my_pass`. The command to process the .mdl file for model Xyzsales.mdl is as follows:

```
cogtr -c -s -mXyzsales.mdl -ksal_log=corpadm/my_pass
```

When you use this command, the User ID and password appear in plain text. See the following topic for information about how to avoid plain text passwords.

Recommendation - Take Steps to Avoid a Plain Text Password When Using the -k Option

If you use the `-k` option to pass user IDs and passwords to the Transformer command line, you must take precautions to avoid a plain text password.

Breaches can occur if unauthorized users are able to invoke the `ps` command or can gain access to the detailed processing information in the log file.

To avoid plain text passwords, we recommend that you adopt one of the following strategies:

- Create a text file containing the `-k` option and one or more user IDs and passwords. Store the file in a secure directory, and ensure that your command line script can easily invoke it.

  For example, you can create the Sal_iqd.txt file to contain two user IDs and passwords:

  ```
  -ksal_log=corpadm/my_pass -ksal_cube=corpis/bld_cube
  ```

  You can then call this secured file from the command line. The following is a UNIX example:

  ```
  cogtr -c -s -mXyzsales.mdl ‘cat Sal_id.txt‘
  ```

- Add MDL statements to the end of the .mdl model file that embed the user ID and password information and update the signon information needed to log on to the database. Then, run cogtr on UNIX using the `-m` option, specifying your modified .mdl file.

  For example, embed the authentication information using the following MDL statements:

  ```
  SignonUpdate "sal_cube" PromptForPassword False UserID "corpis" Password
  "bld_cube"
  ```

- Create a secured but temporary MDL script on the server to update the model signon. Then, run cogtr on UNIX using the `-m` option, specifying your modified .mdl file.

  For example, create the following MDL script, for batch processing using the `-m` option:

  ```
  OpenPY "Xyzsales.pyj"SignonUpdate "sal_cube" PromptForPassword False
  Password "bld_cube"SavePY "Xyzsales.pyj"
  ```

For more information about security, see the IBM Cognos 8 *Administration and Security Guide*.

-l option

This option specifies the user authentication information for IBM Cognos 8. It supplies the signon information needed to authenticate users in one or more security namespaces. The `-l` option provides an alternative to storing the information in the Signons list.

The syntax for using the `-l` option is as follows:

```
cogtr -lsignon=userid/
password -pfilename.py?|-m
filename.mdl
```

The following example opens the Field3.mdl file and grants access to the required namespace using the signon named `field`, user ID `CarlosR`, and password `pw462`. Assuming that the signon matches what is defined in the model, the process runs in batch mode, creating the cubes as defined in the model.

```
cogtr -n -lfield=CarlosR/pw462 -mField3.mdl
```

IBM Cognos 8 can be configured to use authentication to an external namespace, where users are prompted for credentials as part of the IBM Cognos 8 logon process.

You can create Cognos 8 signons to build cubes in batch mode in this environment. This signon maintains the user ID, password, and associated namespace. You create as many Cognos 8 signons as the number of IBM Cognos 8 namespaces your users need to log on to. To enable Transformer to use the Cognos 8 signon automatically, enable the Set As Auto Logon property.

To expand on that, if a user has created many Transformer Cognos 8 signons with Set As Auto Logon property checked, and then saved the model, then when using the command line, by default Transformer will authenticate against all those Transformer Cognos 8 signons, without the need to specify the `-l` option at all.

-m option

This option specifies the .mdl-format model or script file to be processed.

If you use multiple occurrences of `-m`, files are processed in the order of their occurrence.

The syntax for using the `-m` option is as follows:

```
cogtr -mfilename.mdl
```

The following example lists the steps for processing a sample model using the `-m` option:

1. Create a separate file, Savemdl.mdl, that contains the line `SaveMDL "Xyznew.mdl"`.
2. For the file Xyzsales.mdl, use the following command:

   ```
   cogtr -mXyzsales.mdl -mSavemdl.mdl
   ```

3. For the file Xyzsales.pyj, use the following command:

   ```
   cogtr -pXyzsales.pyj -mSavemdl.mdl
   ```

-n option

This Windows-only option opens Transformer in interactive or batch mode.

- –n opens Transformer and runs it interactively.

- –n1 opens Transformer in batch mode with the Transformer application window minimized. There is no space between -n and the display_state argument, 1.
- –n2 opens Transformer in batch mode with the Transformer application window hidden. There is no space between -n and the display_state argument, 2.

In interactive mode, the Transformer application window opens and the user is prompted for information, such as logon credentials, in the same way that they are prompted when Transformer is opened directly. In batch mode, the user is not prompted for information when Transformer runs, even when the information is required. This is expected behavior.

The syntax for using the `-n` option is as follows:

```
cogtr -ndisplay_state -p
filename.py?|-mfilename.mdl
```

The following example opens the model file, Roofing.mdl, without displaying an application window, creates its defined cubes, and then saves the model file.

```
cogtr -n2 -mRoofing.mdl
```

The following example opens the model file, Roofing.mdl, and creates its defined cubes:

```
cogtr -n -sRoofing.mdl
```

-nologo option

This Windows-only option prevents the display of the Transformer splash screen.

The syntax for using the `-nologo` option is as follows:

```
cogtr -nologo -pfilename.py?|-m
filename.mdl
```

The following example opens the model Roofing.mdl without first showing the splash screen:

```
cogtr -nologo -mRoofing.mdl
```

-o option

On Windows, this option regenerates the categories after a model load, but disables cube creation. On UNIX and Linux, it loads the model file, but turns off population of the model with data and cube creation.

On Windows, use this option in combination with `-n` to open Transformer, perform the specified action in batch mode, and then close Transformer.

The syntax for using the `-o` option is as follows:

```
cogtr -o -pfilename.py?|-m
filename.mdl
```

The following example opens the model file, Roofing.mdl, saves any changes to the structure, such as new categories, then closes Transformer without creating any cubes defined in the model.

```
cogtr -s -n -o -mRoofing.mdl
```

The following UNIX/Linux example loads the file, go_sales.mdl, but disables both population of the model and creation of cubes:

```
cogtr -o -mgo_sales.mdl
```

-ox option

This Windows-only option is not a combination of `-o` with `-x`, but is similar to the functionality of the basic -o option on UNIX and Linux, disabling both category creation and the cube rebuild.

The syntax for using the `-ox` option is as follows:

```
cogtr -ox -pfilename.py?|-m
filename.mdl
```

The following example loads the file, go_sales.mdl, but disables both population of the model and creation of cubes:

```
cogtr -ox -mgo_sales.mdl
```

-p option

This option opens the checkpoint file (.qy? on Windows) or loads a binary model file and processes it, beginning from the last checkpoint saved in the checkpoint file if this exists, or beginning from the start of the .py? file. All changes are saved on termination.

This option is not valid with an .mdl file.

Note: A checkpoint file exists if model processing was suspended. The file has a .qy? extension, where the question mark (?) is replaced by the character that is used in your version of Transformer.

The syntax for using the `-p` option is as follows:

```
cogtr -pfilename.py?
```

The following Windows example opens the checkpoint file associated with model file, Sales.pyj, then continues to process the model in batch mode, creating all defined cubes:

```
cogtr -n -pSales.pyj
```

The following UNIX example starts Transformer, process the MDL verb commands in the file monthly_update.mdl, obtains preference settings from the file, trnsfrm_prd.prf, and saves the model:

```
cogtr -pgo_sales_jan.pyj -mmonthly_update.mdl -ftrnsfrm_prd.prf
```

-r option

This option sets the degree of detail for messages written to the log file.

Each level includes the errors and messages for the higher levels. No spaces may appear between `-r` and its `log_level` argument, which is assigned a value from 0 to 4, as follows:

- 0 - the Enable Message Logging box is cleared; logging is suppressed
- 1 - includes only severe errors
- 2 - includes error messages and level 1 messages
- 3 - includes warning messages, and level 1 and 2 messages
- 4 - includes all message levels, from informational to severe; the default setting

The syntax for using the `-r` option is as follows:

```
cogtr -rlog_level -p
filename.py?|-mfilename.mdl
```

The following example opens the model file Roofing.mdl, sets the degree of detail for messages to 2, then runs the process in batch mode to create all defined cubes. The messages are written to F:\Test\Roof.log.

```
cogtr -n -r2 -dLogFileName=Roof.log -dLogFileDirectory=F:\Test Roofing
```

-s option

This option, after successful creation of a cube, saves the model with any new categories added during the category generation process, then closes Transformer.

Do not use this option with `-i` or `-p`. On Windows, you must use this option in combination with `-n`.

The syntax for using the `-s` option is as follows:

```
cogtr -mfilename.mdl -s
filename.py?|filename.mdl
```

where parameters for `-s` are optional.

The following Windows example opens the model file Roofing.mdl, creates its defined cubes, saves the model file, and closes Transformer.

```
cogtr -n -s -mRoofing.mdl
```

The following UNIX example starts Transformer, parses a text model file (.mdl), and saves the changes in a binary model file (.py?).

```
cogtr -mgo_sales.mdl -sgo_sales.pyj
```

-t option

This option sets the current period for the purpose of calibrating relative time calculations. It is equivalent to manually defining a current period on the Windows interface, after clearing the **Automatically Set Current Time Period** box on the **Time** tab of the **Dimension** property sheet.

On Windows, you must use this option in combination with `-n`.

No spaces may appear between `-t` and its argument, and if the category contains hyphens or space characters (as in the sample date range below), you must enclose it in double quotation marks. Also, the `category_code` portion of the command is case-sensitive. This identifier must exactly match the category code in the model.

The syntax for using the `-t` option is as follows:

```
cogtr -tcategory_code -p
filename.py?|-mfilename.mdl
```

The following example opens the model Year3.mdl, sets the current period to the category which has a category code of 20061201-20061231, then runs the process in batch mode to create all defined cubes.

```
cogtr -n -t"20061201-20061231" -sYear3.mdl
```

-u option

This option writes the partition information for a specified cube to the log file, using the following format:

```
datetime Cube
[powercube_name]|[powercube_group_name/
child_cube_name] partition report Partition#
Category Code Category Name Partition Size
```

You must generate cubes before their partition status can be reported. If the command line includes options to generate categories and create cubes, those options are processed before the partition information is obtained.

Partition information cannot be provided for cubes and cube groups to which security has been applied. To obtain the partition information of cubes in a cube group, you must specify the individual cube names, not the cube group name.

The cube name is case-sensitive; for example, you must type uNorth not unorth, for the example below to be valid. No spaces can appear between the option and its argument.

The syntax for using the `-u` option is as follows:

```
cogtr -u[powercube_name]|[
powercube_group_name/child_cube_name] -p
filename.py?|-mfilename.mdl
```

The *powercube_name* parameter represents the name of the cube in the Transformer model. If *powercube_name* specifies a cube group that is not a time-based partitioned cube, all child cubes of the cube group are copied to the deployment location and activated. If *powercube_name* specifies a time-based partitioned cube, it is processed as a single cube; member cubes are not deployed individually.

If a name conflict exists with a child cube of a cube group that has the same name as another cube in the model, use the *powercube_group_name/child_cube_name* parameters.

This Windows example opens the model file Roofing.mdl, discarding a checkpoint file if one exists. It then creates all defined cubes and writes the partition information of cubes North and East to the log file F:\Roof.log.

```
cogtr -i -n -uNorth -uEast -dLogFileName=F:Roof.log -mRoofing.mdl
```

-v option

This option specifies how many data source records to use to create a test cube. If you have a large data source file, this option enables you to do a test run on a limited number of records before processing the entire file.

If the number of records you specify for the test is greater than the total number of records in the source file, the process runs normally, using the entire file.

Because the option syntax does not reference a model file, `-v` is used in combination with `-c`, `-m`, or `-p`, each of which does reference a model file.

The syntax for using the `-v` option is as follows:

```
cogtr -vdata_subset_number
```

The following example processes a subset of the records in the binary model file Xyzsales.pyj (525 records), generating categories, and creating the test cube.

```
cogtr -c -pXyzsales.pyj -v525
```

-x option

This option updates the column and measure scales of the MDL model, provided the data source can handle queries about scale. Therefore the option is supported for relational data sources, but not ASCII or other flat-file data sources.

All column scales are first checked to confirm that they match those in the source. Then, the associated measures are checked and their output scales are updated, as required.

The syntax for using the `-x` option is as follows:

```
cogtr -x -mfilename.mdl
```

The following Windows example opens the Field3.mdl file and grants access to the required database using the signon named `field`, user ID `CarlosR`, and password `pw462`. Assuming that the signon matches what is defined in the model, the process runs in batch mode, updates scales of columns defined in the model, and saves it back to the model.

```
cogtr -n -x -o -s -kfield=CarlosR/pw462 -mField3.mdl
```

-y options

This option specifies how IBM Cognos Series 7 user-class security conversion is performed.

You can use the `-y` options alone to save changes to the model file, or you can use it in combination with other options.

The `-y` options correspond to the three security import options in the **Import model with IBM Cognos Series 7 user class view** dialog box. For more information, see "Upgrade an IBM Cognos Series 7 Secured PowerCube" (p. 183).

-y1 option

Choose this option when you want to maintain the view operations applied in the user class views and use the IBM Cognos Series 7 user classes.

During the transition from an IBM Cognos Series 7 namespace to an alternate security provider, you can use the PowerCube property **All applicable namespaces** to associate all applicable namespaces during migration testing. When you associate all the applicable namespaces to the cube, you can ensure that the group, role, or user dimensional filtering is consistent with that which had been applied for the IBM Cognos Series 7 user class. This option is supported only for migration testing, and cannot be used to deploy cubes in production environments.

The -y1 option requires that you configure the IBM Cognos Series 7 security on which the upgraded model was designed as an available namespace in IBM Cognos 8. The unique identifier that locates the user class in Access Manager is converted to a IBM Cognos 8 identifier, and this process will fail if you use this option with a different IBM Cognos Series 7 namespace.

The following example

- runs cogtr.exe in batch mode, does not generate cubes or update categories
- saves the .mdl file
- preserves the IBM Cognos Series 7 user class views and user classes associated with the IBM Cognos Series 7 model

- logs onto the GO namespace with the Administrator user name and no password

```
cogtr.exe -n2 -ox -s -y1GOnamespace=Administrator/ -mNationalV7.mdl
```

-y2 option

This option imports only the IBM Cognos Series 7 user class views associated with the upgraded model.

Choose this option when you want to import the IBM Cognos Series 7 user class views associated with the model, but not the user classes.

This option allows you to maintain the view operations applied in the IBM Cognos Series 7 user class views but not use a IBM Cognos Series 7 namespace with the custom views, or if you do not intend to expose IBM Cognos Series 7 as an a available namespace configured in IBM Cognos 8.

The following example

- runs cogtr.exe in batch mode
- does not generate cubes or update categories
- saves the .mdl file
- preserves the IBM Cognos Series 7 user class views associated with the IBM Cognos Series 7 model

```
cogtr.exe -n2 -ox -s -y2 -mNationalV7.mdl
```

-y3 option

This option discards the IBM Cognos Series 7 user class views and user classes associated with the upgraded model.

Choose this option when you plan to create new custom views and use only the security objects currently configured in the IBM Cognos 8 namespace.

The following example

- runs cogtr.exe in batch mode
- does not generate cubes or update categories
- saves the .mdl file
- discards the IBM Cognos Series 7 user class views and user classes associated with the upgraded IBM Cognos Series 7 model

```
cogtr.exe -n2 -ox -s -y3 -mNationalV7.mdl
```

Backward Compatibility

Transformer version 8.x supports previous versions of Windows and UNIX/Linux preference files.

Sample Windows Preference Files for IBM Cognos Series 7

Transformer version 8.x command line preference files for Windows are the same as those for previous versions with one exception: IBM Cognos Series 7 Windows files had a `PowerPlay Transformer` section preceding the preferences list. The preference format was `Name=Value`.

```
[PowerPlay Transformer]
CustColor0=0
CustColor1=0
CustColor2=0
CustColor3=0
CustColor4=0
CustColor5=0
CustColor6=0
CustColor7=0
CustColor8=0
CustColor9=0
CustColor10=0
CustColor11=0
CustColor12=0
CustColor13=0
CustColor14=0
CustColor15=0
CubeSaveDirectory=
logging_frequency=-1
DataWorkDirectory=
ModelWorkDirectory=
WindowsSortRule=0
WindowsDateFormat=0
ChildRatioThreshold=35
rowCheckMax=600
sampleMax=300
DetachDataSource=0
LunarFiscalLabeling=0
AutoAutoDesign=1
DlgTitle=3
MaxTransactionNum=50000
EnableDrillThrough=0
LogDetailLevel=4
LogFileDirectory=
LogFileName=
LogFileAppend=0
SourceCharacterSet=3
IncMsgDisable=1
WorkFileMaxSize=1500000000
WorkFileCompress=0
PartitionSizeOverride=0
AutoPartitionOff=0
AutoEdit=0
WinForGndClr=0
WinBacGndClr=16777215
LevelSClr=12632256
LevelRClr=16777088
CatSClr=65280
CatRClr=12632256
LineClr=8421504
MapLiteClr=8454143
MapDarkClr=6019839
MapContstClr=255
MapAllocClr=11468718
DiagScaleFont=0
DiagFont=
DiagFontWeight=0
DiagFontCharset=0
ModelSaveDirectory=
DataSourceDirectory=
DiagFontItalic=0
FullAnimation=1
MultiFileCubeThreshold=0
Introduction_On=1
DsPreviewShowByDefault=0
DsPreviewSampleSize=100
DsPreviewColumnWidthRule=34
OrderCategoriesByDefault=0
WorkFileSortBufferSize=8000000
WorkCountOn=0
DumpCSVPath=
EnablePCOptimizer=1
maximize=1
xOffset=110
yOffset=145
width=768
height=530
```

Sample UNIX and Linux Preference Files for IBM Cognos Series 7

Transformer version 8.x command line preference files for UNIX and Linux are the same as those for previous versions. The following is an example of a transformer.rc file:

```
LogFileDirectory=./../logs
ModelSaveDirectory=/tmp
DataSourceDirectory=/tmp
CubeSaveDirectory=/tmp
DataWorkDirectory=/tmp
ModelWorkDirectory=/tmp
MaxTransactionNum=500000
LogDetailLevel=0
UseTransDAPipe=0
LogFileName=
LogFileAppend=FALSE
LoggingFrequency=-1
WindowsDateFormat=0
MdcDebugOn=
DatDebugOn=0
WorkCountOn=0
DumpCSVPath=
ChildRatioThreshold=35
DetachDataSource=TRUE
FilenameVariables=FALSE
IncUpdateWarnings=TRUE
LunarFiscalLabeling=FALSE
OrderByCategoryLabeling=FALSE
ServerVerbOutput=1
DefaultSeparator=,
ThousandSeparator=,
DecimalPoint=.
ServerWaitTimeout=10
ServerWaitPeriods=30
ServerAnimateTimeOut=3
ServerSyncTimeOut=-1
PowerGridBlockSize=16384U
WorkFileCompress=0
PartitionSizeOverride=0
AutoPartitionOff=0
WorkFileMaxSize=1500000000
WorkFileSortSize=8000000
EnablePCOptimizer=TRUE
TransdaPath=
TransdabPath=
CenturyBreak=20
KeepDataFiles=1
LoaderInterval=
LoaderTimeOut=
LoaderFileSize=
MultiFileCubeThreshold=0
HaltOnSecurityError=FALSE
```

IBM Cognos 8 PowerCube Connection Utility

After a PowerCube is rebuilt, you can update the data source connection information using the IBM Cognos 8 PowerCube Connection Utility. This command line utility is located in the *installation_location*/c8/webapps/utilities/PCConn directory.

To launch the IBM Cognos 8 PowerCube Connection Utility, type **PCConn** at the command line.

You can also use automated processes to update the data source connection. For more information, see "Using Automated Processes to Update PowerCube Connection Information" (p. 216).

The IBM Cognos 8 PowerCube Connection Utility supports the following options. Detailed explanations are provided in the subsections that follow.

Option	Meaning
connect	Connects you to a dispatcher.
connections	Lists the dispatchers to which you are connected.
disable	Disables the PowerCube.
enable	Enables the PowerCube.
get	Lists the current data source connection information.
help	Lists the available IBM Cognos 8 PowerCube Connection Utility commands.
set	Sets the Windows or UNIX/Linux path to the data source.
start	Starts the Report and Batch Report services.
state	Confirms that the PowerCube is disabled.
stop	Stops the Report and Batch Report services.

connect

This option connects you to a dispatcher. At the command prompt, type

```
connect server:port
```

where *server* and *port* represent the IBM Cognos 8 server and port number.

To connect to multiple dispatchers, type each server and port number.

connections

This option lists the dispatchers that you are currently connected to in the IBM Cognos 8 PowerCube Connection Utility. At the command prompt, type

```
connections
```

If your IBM Cognos 8 environment is secured and you are prompted for your logon credentials, type your namespace, user ID and password. If your environment has multiple namespaces, you can log on to any or all namespaces. Do not use Anonymous logon credentials.

disable

This option disables the PowerCube so it can be overwritten with a newer version of the cube. At the command prompt, type

```
disable data_source_name
```

where *data_source_name* is the name of your PowerCube data source.

enable

This option enables the PowerCube. At the command prompt, type

```
enable data_source_name
```

where *data_source_name* is the name of your PowerCube data source.

get

This option provides the current data source connection information. At the command prompt, type

```
get data_source_name
```

where *data_source_name* is the name of your PowerCube data source.

The IBM Cognos 8 PowerCube Connection Utility returns the connection information for the data source, and allows you to see the connection string without opening the Administration tool.

help

This option provides a list of available IBM Cognos 8 PowerCube Connection Utility commands. At the command prompt, type

```
help
```

set

This option lists the dispatchers that you are currently connected to in the IBM Cognos 8 PowerCube Connection Utility. At the command prompt, type

```
set data_source_name
```

where *data_source_name* is the name of your PowerCube data source.

If your IBM Cognos 8 environment is secured and you are prompted for your logon credentials, type your namespace, user ID and password. Do not use Anonymous logon credentials.

start

This option starts the Report and Batch Report services. At the command prompt, type

```
start reportService, batchReportService
```

state

This option confirms that the PowerCube is disabled. At the command prompt, type

```
state data_source_name
```

where *data_source_name* is the name of your PowerCube data source.

stop

This option stops the Report and Batch Report services. At the command prompt, type

```
stop reportService, batchReportService
```

Appendix B: Troubleshooting

This section describes issues and limitations that may be encountered by Transformer users working in a mixed PowerPlay/IBM Cognos 8 environment, with suggested workarounds.

Accessing Error Message Help

Error message help is provided in cases where the basic error message is not sufficient to resolve the problem. The purpose of these help topics is to provide more detail about the cause of the problem, often an example of how the problem manifests itself in the Windows interface, and a possible solution or workaround.

If you are working on the Windows interface when a process fails or when Transformer encounters an invalid entry or request, one or more error messages usually appear.

- ❑ You can click the **Help** button in the message window, where available, to open an Error Message help file, which you can search by TR number.
- ❑ You can also access these topics, organized by TR number, from the index.

Known Issues When Modeling in IBM Cognos Transformer

Documented in this section are known issues and limitations when modeling in IBM Cognos Transformer.

BAPI Error Occurs After the Prompt Specification File Edited Manually

In Transformer, you create a prompt specification for a SAP-based package. You edit the prompt specification file, prompt.xml, manually and save your changes. After editing the file, you attempt to generate a PowerCube using the command line options, for example,

```
cogtr -fpromptspecfilename -n cubename.mdl
```

but the PowerCube is not generated and you receive a BAPI error.

The error is caused by an invalid member unique name (MUN). Because editing the prompt.xml file manually is error prone, we recommend that you do not edit the prompt.xml file manually but create an alternate prompt specification instead.

582131

Unable to Access an IQD Data Source using a Sybase Database Connection

In Framework Manager, you use an IQD file to externalize a model using a Sybase database connection. When you attempt to import the data source file into Transformer, you receive the following error message:

[TR1907] Transformer cannot gain access to database *database_name* with signon information <user ID, password>.

The database connection fails because quotes are added to the SQL query when the data source is created in Framework Manager.

To successfully connect to the IQD data source and import the model, you must first edit certain configuration files in the CS7Gateways\bin directory.

Steps

1. Open the cs7g.ini file and ensure the database type in the connection string is CT, not CT15.

   Cs7g.ini is located in the <*installation_location*>\cognos\c8\CS7Gateways\bin directory.

2. In the [Services] section, include the following:

   ```
   CTDBA=ctdba, ctdba15
   ```

3. Save your changes.

4. Open the cogdmct.ini file and in the [UDA USER OPTIONS] section, specify the following:

   ```
   Attach=SET QUOTED_IDENTIFIER ON
   ```

   Cogdmct.ini is located in the <*installation_location*>\cognos\c8\CS7Gateways\bin directory.

5. Save your changes.

6. Open Transformer and import the data source.

   588853

Importing Time Dimensions from IBM Cognos 8 Packages Not Supported

When you create a query in Transformer based on a Framework Manager package that contains hierarchical time-related categories, Transformer interprets the time-related categories as a regular dimension and not as a time dimension. As a result, the time dimension in your PowerCube will not contain any relative time categories.

Ensure that you import all of the data needed to define your time dimension, and use Transformer to create the date levels and categories.

Data in Multiple Languages Does Not Display Properly

When you create a data source using an IBM Cognos 8 report, and the report includes data in multiple languages, some of the language characters do not display properly in the Transformer Data Source Viewer. These characters are displayed as -- .

Transformer does not support reports with multilingual data as a data source. When the operating system locale is properly set, Transformer displays the characters for that locale.

There is no workaround for this problem.

Unable to Use an IQD Created in Framework Manager That Contains an Oracle Stored Procedure

In Transformer, when trying to open an IQD created in Framework Manager that contains an Oracle stored procedure, you may receive a message similar to the following:

(TR0118) Transformer can't read the database [datasource] defined in <Lan location>\ <datasource><iqd_name>.iqd.

DMS-E_General A general exception has occurred during operation 'execute'

The native SQL generated in an IQD created in Framework Manager is wrong. The IQD cannot be used in Transformer.

To resolve this problem, execute the stored procedure in Framework Manager and set the **Externalize Method** to **IQD**. Create a model query subject from the executed stored procedure, then publish the package and open it in Transformer.

Preventing Errors When Model Calculations Use Double Quotation Marks

If you try to open an .mdl-format model containing calculations that include double quotation marks, as might be used to create concatenated categories, you may get an error, even if you followed the recommended practice of wrapping these calculations in single quotation marks.

This is because .mdl-format models do not support the use of single and double quotation marks together, if the ObjectIDOutput flag is set to True, which is the default model creation setting.

To avoid this problem, you have two choices:

- You can open the cogtr.xml.sample file in a text editor, search for the string ObjectIDOutput, and change the setting to 0. Save the cogtr.xml.sample file as **cogtr.xml**. Restart Transformer, and resave the model.
- You can use your RDBMS or a tool such as Framework Manager to perform the required calculations, and then import the data into your model.

Whichever strategy you choose, you can then open the .mdl or py?-format model without error.

537305

Framework Manager and Transformer may Display Different Locale Session Parameters for Some Languages

Transformer version 8.x may not return data in the expected locale during test or cube build when the following conditions are encountered:

- The locale shown in the File/Session information in Transformer is not included in the Framework Manager parameter map for session parameters.
- The modeler attempts to create a data source in Transformer using a query subject from the package where the locale does not exist.

When this is encountered, the locale of the modeler’s session parameter does not exist in the Framework Manager parameter map. As a result, the data returned will not be the locale of the Session information shown in Transformer.

To avoid this problem, add the locale string that is displayed in the Transformer File/Session information to the Framework Manager parameter list so that Transformer can retrieve the expected data when accessing the data source. However, the model metadata will still be shown in English, or in the Framework Manager design language.

Regular Columns cannot be Converted to Calculated Columns and Vice Versa

When you attempt to convert a regular column to a calculated column by opening the **Column** property sheet, the **Calculated** button is unavailable.

In Transformer version 8.x, you can no longer convert an existing regular column to a calculated column by changing the column properties. Similarly, existing calculated columns cannot be converted into regular columns by changing the column properties.

In Transformer version 8.x, you can only create calculated columns using the **Insert Column** feature. For more information, see "Define a Calculated Column" in the Transformer *User Guide*.

This change in functionality does not affect how calculated columns are imported from an IBM Cognos Series 7 model into Transformer version 8.x. Existing calculated columns originally created in IBM Cognos Series 7 will be imported correctly.

Transformer Takes a Long Time to Retrieve Data from an SAP-based Data Source

You are attempting to retrieve data from an SAP-based data source in Transformer with null suppression turned off. The retrieval takes a long time to complete.

Ensure that the machine where Transformer is installed has sufficient memory to perform the import. If physical memory is limited, Transformer may perform the operation very slowly. In this situation, you can end the task using Task Manager.

589874

Categories Missing When Creating a Transformer Model Based on an SAP Query Containing a Manually Created SAP Structure

The stream extract interface that reads the fact data doesn't handle certain features of the SAP queries. A manually created structure in the query will look like a dimension when Transformer completes the import from the Framework Manager package, but incomplete data is returned. A dimension added to the SAP query as a characteristic will look like a dimension but no data is returned.

If you must use a BEx query with these limitations, consider turning off the stream extract and rely on an MDX query. Note that if the MDX query is large, it may fail.

591834

Error Occurs When Creating a PowerCube Containing an SAP Unbalanced Hierarchy

You import an SAP package into Transformer that contains an unbalanced, ragged hierarchy and you receive a TR2317 error when you create a PowerCube.

To avoid this error, before generating categories for the dimension, do the following steps.

Steps

1. In the **Dimension Map**, right-click the lowest level in the ragged unbalanced hierarchy that is marked unique.

2. Click **Properties** and on the **Source** tab, click **Move**.

3. From the **Run** menu, click **Generate Categories**.

4. From the **Run** menu, click **Create PowerCubes**.

   589899

Rebuilding a PowerCube Soon After Publishing Produces a TR0787 Error

After publishing a PowerCube using the **Publish** wizard in Transformer, the PowerCube file is locked for a few minutes by the IBM Cognos 8 server. If you attempt to rebuild the cube during this time, the cube build may fail, with Transformer error TR0787 indicating that the cube is being used by another application.

To avoid this situation, do one of the following:

- Do not use the **Publish** wizard to publish the cube.

- Wait for the file lock to be released, and then rebuild the cube.

- Build the cube in a location that is different from the location where the cube is published.

596414

Known Issues Using Cubes in the IBM Cognos 8 Studios

Transformer version 8.x enables PowerCubes and their data sources to be published directly into the IBM Cognos 8 environment, without requiring Framework Manager as an intermediary.

Documented in this section are known issues and limitations when viewing PowerCubes created for the IBM Cognos 8 Web Studios, such as Analysis Studio and Report Studio.

Not Yet Optimized IBM Cognos PowerCubes May Open Slowly in IBM Cognos 8

If PowerCubes created with previous versions of Transformer take too long to open in the IBM Cognos 8 Web studios, we recommend that you run a command line utility named pcoptimizer, supplied with IBM Cognos 8, to improve run-time performance. This optimization utility is suitable for older PowerCubes when the model no longer exists or the data used to build the PowerCube is no longer available. It is not necessary to run this command line utility for cubes created in Transformer version 8.x.

Steps

1. Back up your target PowerCube, then navigate to the *Cognos_8_installation_location*/bin directory.

2. On Windows, open a command line window and run PCOptimizer.exe.

3. On UNIX/Linux, enter the following line to run the optimization command line utility:

   ```
   pcoptimizer [-t] [-v] [-h]cubename
   ```

where *cubename* is the fully qualified PowerCube or time-based partitioned control cube name with the .mdc extension, if the PowerCube resides in the same location as pcoptimizer. Otherwise, *cubename* is the full path with the .mdc extension.

Note: This method only supports metadata extraction. To set up user-configurable drill-through, you must use Transformer. Wildcard character support is not currently available. You must therefore invoke the utility once per PowerCube. If *cubename* is not provided, the program enters an interactive mode, prompting you for a PowerCube name and accepting keyboard input. The optional parameters are as follows:

- `-t` or test mode; it tests whether the metadata was extracted and loaded into the PowerCube. The return code indicates the status.
  - 0 if the metadata was extracted and loaded
  - 10 if the metadata was not loaded
  - 20 if an error occurred while accessing the PowerCube
- `-v` or verbose mode; text is output to standard output (stdout), indicating what was done, including any error messages. If running in interactive mode, `-v` is assumed. All text is output in English only.
- `-h` for command-line help; if *cubename* is not provided, it prints the usage and options to the screen.

Analysis Studio Shows the Wrong Currency Symbol

When published to IBM Cognos 8 Analysis Studio, PowerCubes show a default currency rather than the currency associated with the locale of your servers and PCs. For example, GBP (£) is shown as $.

To resolve this problem, you can do one of the following:

- Create a currency table when you prepare your model in Transformer and embed a default currency symbol into the resulting PowerCubes, based on the system locale used by your Transformer computer.
- For PowerCubes that do not contain an embedded currency table, set the `fallbackCurrency` parameter as the default currency.

Steps to Create a Currency Table in Transformer

1. In Transformer, from the **File** menu, click **Currency Table** and click **OK**.
2. Right-click each currency measure and click **Allow currency conversion**.

This default currency table does not include currency information for any locales other than your running locale. Also, you cannot convert to a different currency while working in Analysis Studio.

Step to Set fallbackCurrency as the Default Currency

- Define a default currency by setting the `fallbackCurrency` parameter in the *installation_location*\configuration\qfs_config.xml file to GBP (Great Britain Pounds) or to an alternative currency code as listed in the *installation_location*\bin\ccli18nrescr_xx.xml file.

  Here is an example.

```
<!-- execution layer providers-->

<provider name="PowerCubeODP" libraryName="pcodp"
connectionCode="PC">

    ...

    <providerDetails>

        <parameters>

           <!-- Max depth of nested calculated members
within a query. -->

           <parameter name="maxCalculatedMemberNestingDepth"
value="30"/>

           <!-- Normalize yen/won currency symbols
- set to "false" to disable -->

           <parameter name="normalizeYenWon" value="true"/>

           <!-- Fallback currency for cubes with no
default currency specified - set to USD, EUR etc. -->

           <parameter name="fallbackCurrency" value="USD"/>

        </parameters>

    </providerDetails>

</provider>
```

Changes to Decimals in Currency Formats

When you open a PowerCube in an IBM Cognos 8 Web studio or in IBM Cognos 8 Business Intelligence Mobile Analysis version 8.3, you may notice changes in the number of default decimal places shown in currency formats.

This behavior is due to the following changes:

- The default decimal formatting in currency formats is now determined by the measure format selected in the cube, instead of from the data source currency table definition.

  For example, if the Actual Revenue measure format specifies two decimal places and the USD currency in the currency table specifies no decimal places, two decimal places will appear in the USD currency value.

- Calculations that include a division operator and at least one currency operand will now show a resulting value with three decimal places only when

  - neither of the currency values includes decimals

  - two currency operands have different numbers of decimal places

In all other calculations of this type, the number of decimals in the resulting value is determined by the number of decimals in the currency value. The following examples illustrate this new behavior:

- $4.00 / $2.00 = $2.00
- $4 / $3.0000 = $1.3333
- $4 / $3 = $1.333
- $4.0 / $3.00 = $1.333

Ragged or Unbalanced Hierarchies Result in Unexpected Behavior

In ragged or unbalanced hierarchies, some members that are not at the lowest level of the hierarchy may have no descendants at one or more lower levels. Support for these hierarchy gaps in relational sources is limited. For OLAP sources, more complete support is provided, but some reports may result in unexpected behavior:

- Groups corresponding to missing members may appear or disappear when grouped list reports are pivoted to a crosstab. This happens with set expressions using the filter function, and detail filters on members.
- Ragged and unbalanced sections of the hierarchy are suppressed when set expressions in that hierarchy are used on an edge.
- When a crosstab is sectioned or is split into a master-detail report, sections corresponding to missing members become empty.

Some of these behaviors may be corrected in a future release, while others may be codified as supported behavior. To prevent these behaviors, avoid the scenarios above.

The following scenarios are believed to be safe:

- one or more nested level references on an edge, with no modifying expression.
- a hierarchy reference on only one level of one edge.
- one or more explicit members or sets of explicit members as siblings on only one level of one edge.
- summaries of the previous three scenarios.

In all cases, reports based on ragged and unbalanced hierarchies should be tested to confirm that hierarchy gaps are handled correctly.

Error Opening Saved Reports After PowerCube Refresh

After a PowerCube is refreshed, when you open reports that include drill down on directly referenced special time categories, you may receive the following error:

OP-ERR-0025 The following OLAPPlanner internal error occurred: *internal error*

When drill down is not used, you will not receive the error; however, the data shown may not be correct.

There is no workaround for this problem.

Unable to Open Sample Model, Great Outdoors Sales.mdl, and Generate Cubes

If your setup information for the Great Outdoors Sales.mdl is incorrect, you will be unable to open the sample model for Transformer, Great Outdoors Sales.mdl, or generate cubes.

To avoid this problem set up the Great Outdoors Sales.mdl as follows:

1. Modify the Cs7g.ini to contain [Databases] connections.

   The Cs7g.ini file is located in the *installation_location*/c8/cs7Gateways/bin directory.

2. Open ODBC Data Source Administrator and create a new ODBC data source named great_outdoors_warehouse to connect to the SQL server database, GOSALESDW, which is provided with the sample installation.

3. Connect using a valid User Id and password for SQL Server authentication.

4. Open the model.

Data Records that Support External Rollup Measure Data are Not Always Created

When PowerCubes are built, data records that support External Rollup Measure data are not created in certain circumstances.

To receive a software update, access the IBM Cognos Resource Center (http://www.ibm.com/software/data/support/cognos_crc.html). After you apply the update, you can rebuild the PowerCube with the expected results.

579654

Appendix C: Error Messages

This section provides information on error messages in IBM Cognos Transformer.

Accessing Error Message Help

Error message topics are provided in cases where the basic message is not sufficient. Their purpose is to identify the possible cause of a problem, add an example of how it manifests itself on the Windows interface and, where feasible, suggest a solution or workaround.

If you are working on the Windows interface when a process fails, or when Transformer encounters an invalid entry or request, one or more error messages may appear. To learn more about the error messages

- you can click on the **Help** button in the message window, where available, to access this Error Message Help.
- you can access individual topics, listed by their TR number, from the index.

Transformer Error Messages

This section lists IBM Cognos Transformer error messages for which additional help is provided.

TR0104

A user was expected but not found. Either you have referred to a user by an object name or object identifier that Transformer is unable to locate, or you have not referenced the user.

This error occurs when a script that uses Model Definition Language (MDL) requires an object that Transformer cannot find. In this case, the object is the ID for an authenticated user (CAMID).

The problem may be that the object was not referenced, or that it was referenced with an incorrect object name or object identifier.

You can verify Transformer object names and identifiers on the Windows interface. To make them visible, select the **Object name** and **Object identifier** check boxes on the **Titles** tab of the **Preferences** property sheet. The object name and identifier will appear in the title bar at the top of each property sheet, and also in a tool tip when your pointer hovers over an object.

You can also find object identifiers in the .mdl file. For more information, see the Transformer *Developer Guide*.

TR0105

The view of an ancestor custom view restricts the operation you requested.

You have attempted to override a security setting that was imposed at a higher level in the view hierarchy. Security for this object is inherited from the ancestor custom view.

If you require access to a category that is inaccessible due to access controls placed on its ancestor, you can

- remove the restrictions imposed by the ancestor custom view
- create a new custom view at the ancestor level, with the appropriate security
- selectively add users from the lower-level custom view to the ancestor custom view, so that these users can access the required information

For more information, see "Adding Security" (p. 173).

TR0106

Custom views are not allowed to perform the operation you requested.

You have tried to perform an operation that Transformer does not support.

For more information, see "Adding Security" (p. 173).

TR0107

You cannot view the categories in a dimension that an ancestor custom view has restricted.

This error message may appear when you make updates to a model using an MDL script. Under normal circumstances, the Transformer Windows interface prevents you from viewing all of a dimension for which a custom view is applied in an ancestor custom view.

Verify that the MDL script is correct, and that you have specified the correct object identifier for each of the objects referenced.

If descendants of the protected ancestor in the custom view do require access to the entire dimension, you can

- change the properties of the ancestor custom view so that all categories are included

  Note: If you use this option, all members of the subordinate custom view (the one for which you are trying to include the categories) will gain access to the entire dimension.

- add specific users from the subordinate custom view (the one for which you are trying to include the categories) to the relevant ancestor custom view

  However, this will give the users access to all information that is accessible by the ancestor custom view.

- create a new custom view in which the categories are included, adding users to it as required to meet your reporting needs

For more information, see "Adding Security" (p. 173).

TR0108

An ancestor custom view excludes this measure.

This error message may appear when you make updates to a model using an MDL script. Under normal circumstances, the Transformer Windows interface prevents you from including a measure that is excluded in an ancestor custom view.

Verify that the MDL script is correct, and that you have specified the correct object identifier for each of the objects referenced.

If the custom view for which you want to include the measure does require access to that measure, you can

- change the properties of the ancestor custom view in which the measure is excluded

  Note: If you use this option, all members of the subordinate custom view (the one for which you are trying to include the measure) will gain access to the measure.

- add specific users from the subordinate custom view (the one for which you are trying to include the measure) to the ancestor custom view in which the measure is included

  However, this will give the users access to all information that is accessible by the ancestor custom view.

- create a new custom view in which the measure is included, adding users to it as required to meet your reporting needs

For more information, see "Adding Security" (p. 173).

TR0109

Transformer could not read the data source.

You have specified an invalid data source, or a file format that is not appropriate for the type of data source defined in your model.

For example, this error may occur if you are using an .iqd file as a data source with a 32-bit ODBC driver, and the driver setup specifies the wrong transaction level. It may also occur if the required security information is lacking or the tnsnames.ora file references an incorrect SQL *Net connection string.

To resolve the problem, ensure that your source file is valid, and that it uses a format that is appropriate for the type of data source defined in the model.

For example, if you are using an .iqd file with a 32-bit ODBC driver, try changing the transaction level from **Repeatable Read** to **Read Uncommitted**, and enter a user ID and password in the **Client Access** driver setup.

TR0110

No records were found in data source *data_source_name*; the file is empty.

You have tried to generate categories from an empty input file. The file exists, but contains no records that Transformer can use.

One of your source files may have been inadvertently overwritten by an empty file, or the process that creates your source files may have failed and created a file with no records.

To resolve the problem, ensure that each source file associated with the data in your model contains the correct data. You can confirm the name of the file associated with a data source in the **Source** tab of the **Data Source** property sheet.

TR0111

file_name is not a valid file name.

You have specified an invalid file name for a PowerCube or a data file that is related to a data source.

To resolve the problem, check the name of the file you entered, and verify that it is correct.

TR0112

There is not enough memory available.

There is insufficient memory for Transformer to perform some operation.

Solution for Windows

Close as many open applications as possible. If this fails to correct the problem, save your model, exit, and then restart Transformer. You may need to restart Windows.

If this is a recurring problem, you may need to augment computer memory in order to continue working on the model or PowerCube.

Solution for UNIX/Linux

Try lowering the `MaxTransactionNum` setting. You may also need to increase the memory setting on your data server. For information about optimizing UNIX/Linux production environments, see "Guidelines for Optimizing Production Environments" (p. 219).

Note: If you are working on an HP9000 computer, the problem may be the data region setting. Try increasing the `MAXDSIZ` kernel setting. For more information, consult your HP9000 documentation.

TR0113

Transformer cannot create the file *file_name*.

Transformer is unable to create a file.

To resolve the problem, ensure that you are not attempting to overwrite a read-only file, and that the file is not currently locked by another process.

TR0114

Transformer cannot write in the model temporary file. Please check if there is enough free disk space in the temporary directory.

This error usually indicates that there is insufficient disk space for the temporary work files.

To estimate how much disk space you need to create a cube, multiply the size of all data sources by 3.5. To locate the temporary work files, check the **Directories** tab of the **Preferences** property sheet.

Then, to resolve the problem, free up the required amount of disk space on the work file drive and repeat the failed process.

TR0116

Transformer cannot open the file *file_name*.

To resolve the problem, ensure that the path and file name are correctly specified in the **Directories** tab of the **Preferences** property sheet.

If this error occurs when you are running an MDL script from the command line, the problem may be that the script contains the command `SavePY` and you are also using the `-s` command line option to save the file in .py? format. If so, try removing the `-s` option.

TR0117

Transformer cannot open the file *file_name*.

You have tried to generate categories or create a PowerCube, and Transformer is unable to find one of the source files associated with a data source used by the model.

To resolve the problem, ensure that the source files are correctly specified on the **Source** tab of each **Data Source** property sheet, and that you have access to the drives and directories where these files are stored.

If you have specified an alternate source file for the PowerCube on the **General** tab of the **PowerCube** property sheet, ensure that the alternate source file exists.

If you are building cubes in a multiple-cube model, ensure that the source file information is complete and correct for all cubes, including disabled cubes.

TR0118

Transformer cannot read the database *database_name* defined in *PowerCube*.

Transformer cannot open the database specified as the data source.

If you used an .iqd file as the data source in your model, do the following:

- Check the size of the query to see if it exceeds the SQL limit and, if so, try removing some of the columns in the report that the .iqd file is based on.
- Ensure that the appropriate access was granted, and that the database user ID and password are valid.
- Confirm that the report that created the .iqd file runs successfully.

If you are using Transformer to generate a PowerCube on an HP-UX 10 server computer, and the data source is an .iqd file that accesses a Sybase database, ensure that the following requirements are met:

- Open Client must be installed on the HP-UX server computer.
- The Sybase and DSQuery environment variables must be set.
- The Sybase /bin directory must be in the path.

TR0128

Only data source input files of the following types are supported in this mode *data_source_list*.

You are using Transformer on a UNIX/Linux server but are not specifying an accessible data source.

To resolve the problem, specify a data source that your UNIX/Linux server can access, and then rebuild your PowerCube or cube group.

TR0131

The argument for command line option -k is invalid.

When trying to pass database login information to Transformer using the `-k` command line option, an invalid entry has been detected.

To resolve the problem, ensure that you enter a valid value for the `-k` option. For more information, see "Command Line Options" (p. 237).

TR0132

The database name in the command line is blank.

When trying to pass database login information to Transformer using the -k command line option, there is no database name provided.

To resolve the problem, ensure that you enter a valid database name . For more information, see "Command Line Options" (p. 237).

TR0133

Unable to delete file *file_name*.

Transformer is unable to delete a model file.

To resolve the problem, ensure that the file is not currently locked by another process, and that it is not a read-only file.

TR0137

Cube Group *cube_group_name* (and possibly others) contains cubes based upon key orphanage categories. Due to the temporary nature of these categories, the data in these cubes may belong to other cubes within their respective cube groups.

Transformer has created a key orphanage in the target level for a cube group. This can happen in models for which the following conditions exist:

- The model uses multiple data sources.
- The model contains one or more cube groups.
- The data in the specified data sources does not provide sufficient information to connect a category to one of the target categories in the cube group.

In the **PowerCubes** list, Transformer generates a cube definition for the key orphanage category within the cube group. You cannot delete this cube. However, the problem will be automatically corrected if a subsequently processed data source contains data that can provide the path from the orphaned category to the source category in the target level.

Alternatively, you can resolve the problem by adding information to the model that eliminates the need for both the key orphanage and the orphaned cube. This can be done in one of the following ways:

- Manually assign the categories in the orphaned category to one of the others in the target level, assuming you know which one they belong to.

- Determine what data is missing in your original data source, and either supply that data or add a new source with the information needed to complete the path from a category in the target level to the orphaned categories.

If a key orphanage is created and later deleted during a single cube generation process, the resulting cube group will have an invalid status.

Transformer can eliminate a key orphanage when all of its descendants are placed under other categories in other cubes within the cube group. In such cases, you must regenerate the cube group so that the data is moved into the correct cubes. For more information, see "TR0713" (p. 300).

TR0149

A data input conversion or overflow error has occurred.

This error occurs when the data for a measure is too large for the storage type that is specified for it. This can arise if a measure includes decimals, or is assigned a large input or output scale.

To resolve the problem, try to minimize the **Input scale** property, as specified on the **General** tab of the **Column** property sheet, and the corresponding **Output scale**, as specified on the **General** tab of the **Measure** property sheet.

If this does not work for your model, change the **Storage Type** property on the **General** tab of the **Measure** property sheet to a larger storage type.

TR0202

You did not include any dimensions. Transformer cannot create a PowerCube.

You have tried to create a PowerCube in which all the dimensions are omitted.

To resolve the problem, open the **PowerCube** property sheet and click the **Dimensions** tab. Ensure that at least one dimension is not set to **Omit Dimension**.

TR0203

You did not include any measures. Transformer cannot create a PowerCube.

You have tried to create a PowerCube in which all the measures are omitted.

To resolve the problem, open the **PowerCube** property sheet and click the **Measures** tab. Ensure that at least one measure is not set to **Exclude**.

TR0205

Double quotes were changed to single quotes in the source value, label, description, sort value, or some other text field. This action was taken to avoid syntax errors in MDL.

When Transformer generates MDL, it uses double quotes to define the model. This message informs you that double quotes in the source data have been changed to single quotes so that Transformer can distinguish between the source data and the structure of the model.

For more information about using Model Definition Language (MDL) scripts, see the Transformer *Developer Guide*.

TR0206

You did not include any dimensions in PowerCube *cube_name*. The cube was not created.

You have tried to create a PowerCube in which all the dimensions are omitted.

To resolve the problem, open the **PowerCube** property sheet and click the **Dimensions** tab. Ensure that at least one dimension is not set to **Omit Dimension**.

TR0207

Data source *data_source_name* is not related to any dimension so it cannot be processed.

You have tried to generate categories using a data source that is not associated with any level in any dimension in the model. This can happen when you add a data source to the model without specifying that at least one dimension is derived from that data source.

To resolve the problem, associate a data source with one or more dimensions in your model.

TR0208

Consolidation for the PowerCube *cube_name* (and *n* others) is suppressed because a before-rollup calculation was also defined.

When a model includes a calculated measure with a timing of **Before Rollup**, consolidation is suppressed for all cubes containing that measure.

This allows the consolidated records to be written to the cube as specified. If Transformer were to perform its own consolidation pass, there would not be sufficient detail in the PowerCube records to perform the before rollup calculation correctly.

If you want cube records to be consolidated, ensure that the cube does not contain calculated measures with a timing of **Before Rollup**.

TR0209

Sorting and consolidation for the PowerCube *cube_name* (and *n* others) must be performed to determine correct state measure values.

This message informs you that, in spite of any contradictory settings for consolidation in the **PowerCube** property sheet, one of the following model settings dictates that consolidation is required:

- A setting other than **Default** is specified for **Duplicates** rollup.
- A setting other than **Pre-sorted** is specified for consolidation.

TR0210

n data input conversion or overflow errors occurred. See the log file for details.

In one of the data source files for the model, the value specified for a measure is either too large or uses an invalid storage type.

In the log file associated with the model being updated, a line such as the following appears:

```
(TR1703) A data input conversion or overflow error occurred at source record
number 100 for measure 'Revenue' in source file 'path\filename'.
```

Use the logged information to identify which input data records contain invalid source values, and then make the required corrections.

TR0214

The temporary file for source file *source_file_name* and PowerCube *cube_name* is empty. Check that your source file contains data.

During the PowerCube creation process, Transformer has been unable to find data for use in a PowerCube. This can happen if

- your source data file is empty
- you have applied one or more views that exclude all the data for all dimensions in a cube
- the timing for all data sources is such that none of them are used to create PowerCubes
- the data source containing measures for your model is associated with categories in an alternate drill-down path. For more information, see TR1320.
- the source file contains BLOB (Binary Large Object) fields, which Transformer does not support

If your source file is an .iqd file, and you have IBM Cognos Impromptu installed, rerun the report in Impromptu and ensure that data is being returned.

TR0215

The PowerCube *cube_name* (and *n* others) may require consolidation. Consolidation can't be performed because of before-rollup calculations or because a cube is designated as using Direct Create.

This message informs you that Transformer cannot perform consolidation because there are conflicting settings in the model.

For example, consolidation cannot occur if the model contains calculated measures with a rollup timing of **Before Rollup** because the consolidation process would eliminate from the PowerCube the records required to perform the before-rollup calculation.

TR0217

Data source *data_source_name* is turned off for cube generation, and the process has halted.

This message can arise if the model contains structural data sources, and no transactional (or measure) data.

Ensure that the **PowerCube Creation** box on the **General** tab of the **Data Source** property sheet is selected, if you want this data source processed during cube generation.

TR0301

There are no columns for AutoDesign to use.

Transformer is unable to locate columns for use in the **AutoDesign** process.

Ensure that the columns in the data sources for your model have not been deleted.

TR0303

Transformer has detected invalid allocation specifications. These allocations will be removed. Do you want to continue?

A calculated measure was allocated and then deleted from the model.

Choose **Yes** to have Transformer change the allocation on which the measure was based to **N/A**, if the allocation applies to a level, or **Constant**, if it applies to an entire dimension.

TR0404

You didn't specify a server data source. Do you want to continue?

This error occurs if you try to generate categories on a UNIX/Linux server but have not specified a data source that the server can access.

To resolve the problem, specify a data source that resides on, or is accessible by, your UNIX/Linux computer.

TR0408

This model was built with a system language setting of *language_setting*, which differs from the current setting. The language setting dictates the sorting rules used in category ordering. Do you want to resort all ordered categories?

This message appears when a model populated on a computer with one language setting is moved to a computer with a different language setting.

Your PowerCube will still generate successfully, despite this warning message.

TR0412

Error encountered while trying to save preferences to *path_name*\cogtr.xml

This error may be caused by insufficient space on your hard disk or by an inaccessible cogtr.xml file.

If the problem is caused by insufficient disk space, either create more disk space or switch processing to another drive. If the problem is caused by an inaccessible cogtr.xm file, ensure that cogtr.xml is not open or marked read-only.

TR0420

A PowerCube group was expected but not found. Either you have referred to a PowerCube group by an object name or object identifier that Transformer is unable to locate, or you have not referenced the PowerCube group.

This error occurs when a script that uses Model Definition Language (MDL) requires an object and is unable to find it. In this case, the object is a PowerCube group.

The problem may be that the PowerCube group was not referenced, or that it was referenced with an incorrect object name or object identifier.

You can verify Transformer object names and identifiers on the Windows interface. To make them visible, select the **Object name** and **Object identifier** check boxes on the **Titles** tab of the **Preferences** property sheet. The object name and identifier will appear in the title bar at the top of each property sheet, and also in a tool tip when your pointer hovers over an object.

You can also find object identifiers in the .mdl file. For more information, see the Transformer *Developer Guide*.

TR0423

A calculation definition was expected but not found. Either you have referred to a calculation definition by an object name or object identifier that Transformer is unable to locate, or you have not referenced the calculation definition.

This error occurs when a script that uses Model Definition Language (MDL) requires an object and is unable to find it. In this case, the object is a calculation definition.

The problem may be that the calculation definition was not referenced, or that it was referenced with an incorrect object name or object identifier.

You can verify Transformer object names and identifiers on the Windows interface. To make them visible, select the **Object name** and **Object identifier** check boxes on the **Titles** tab of the **Preferences** property sheet. The object name and identifier will appear in the title bar at the top of each property sheet, and also in a tool tip when your pointer hovers over an object.

You can also find object identifiers in the .mdl file. For more information, see the Transformer *Developer Guide*.

TR0476

A category set was expected but not found. Either you have referred to a category set by an object name or object identifier that Transformer is unable to locate, or you have not referenced the category set.

This error occurs when a script that uses Model Definition Language (MDL) requires an object and is unable to find it. In this case, the object is a category set.

The problem may be that the category set was not referenced, or that it was referenced with an incorrect object name or object identifier.

You can verify Transformer object names and identifiers on the Windows interface. To make them visible, select the **Object name** and **Object identifier** check boxes on the **Titles** tab of the **Preferences** property sheet. The object name and identifier will appear in the title bar at the top of each property sheet, and also in a tool tip when your pointer hovers over an object.

You can also find object identifiers in the .mdl file. For more information, see the Transformer *Developer Guide*.

TR0500

You cannot move a category outside its drill-down path.

On the diagram for a dimension that has an alternate drill-down structure, you have tried to move a category from one drill-down path to another.

To avoid this problem, only move categories within the scope of their own drill-down paths.

TR0501

A regular category cannot be the parent of a special category.

Under normal circumstances, the Transformer Windows interface does not allow you to make a regular category the parent of a special category. However, this error message may appear if you try to perform this action using an MDL script.

To resolve the problem, ensure that the ID numbers used in the MDL script refer to objects for which the stated action is valid.

TR0502

You must specify a category code when you create a manual category.

You have tried to create a category manually, in a manual level, and you have omitted the category code. By default, Transformer will try to assign a unique category code to each of the categories you create.

This category code is displayed in the **Category code** box on the **General** tab of the **Category** property sheet. Do not delete this category code.

TR0503

Another category already has this category code. Each category requires a category code that is unique within the dimension.

When creating or modifying a category manually, in a manual level, you have changed the category code so that it conflicts with an existing category code. By default, Transformer will try to assign a unique category code to each of the categories you create.

This category code is displayed in the **Category code** box on the **General** tab of the **Category** property sheet. If you modify this code, ensure that it does not conflict with an existing category code.

TR0504

You must specify a source value for a category in a source level. You can do this from the **Source** tab in the **Level** property sheet.

You may have inadvertently deleted the source value associated with a category. Source values are required for categories in all source levels. Categories in manual levels derive their source values from their category codes.

To resolve the problem, edit the source value on the **Category** property sheet to provide the correct value, or delete the category and allow Transformer to regenerate it.

TR0505

This action would result in two categories with the source value *source_value* under different parent categories in level *level_name*, which is designated as unique. The action is cancelled.

You have tried to specify that the categories within a level are unique when Transformer has verified that they are not. This can occur when you

- create an alternate drill-down structure and allow Transformer to designate the convergence level as unique

- change the source value of a date category in such a way that two identical date categories appear in the same level
- select the **Unique** option on the **Source** tab of the **Level** property sheet

The category values in a convergence level that connects multiple drill-down paths must be unique. Similarly, in order for Transformer to be able to directly relate source column values to categories in a level, the category values in that level must be unique.

If a level is at the convergence of an alternate drill-down path, or if your data source contains a column for a level that lacks the ancestor levels needed to provide its context, you must ensure that each category in that level is unambiguously identifiable by its value alone.

For example, suppose the convergence level, City, is found in a Regions dimension that contains the levels Country, State, and City. To avoid uniqueness problems, ensure that your source data qualifies all identically named cities so they can be unambiguously identified. For example, the city Burlington is found in the states Massachusetts and Vermont, so you modify your source to add qualifiers for the names Burlington-MA and Burlington-VT.

TR0507

The share reference *category* is not a category in the same dimension.

You have tried to apply a share using a level or category that is in another dimension.

To avoid this problem, when you set up a share, make sure you set it to a level or category in the same dimension as the categories to which you apply the share.

TR0508

You cannot move a category into or out of a convergence level.

On the diagram for a dimension with multiple drill-down paths, you have tried to move a category either out of or into a convergence level.

To avoid this problem, make sure that you do not move the categories in a convergence level out of that level, or move new categories from other levels into the convergence level.

TR0510

The share category ID is invalid. A share category ID must be the object identifier of another category in the same dimension.

You have tried to specify an invalid share. Each share

- must involve an existing category
- must be relative to a higher level in the same dimension
- must be identified by a share object identifier

To view the object identifier for each object, select the **Object identifier** check box on the **Titles** tab of the **Preferences** property sheet. The object name and identifier will appear in the title bar at the top of each property sheet, and also in a tool tip when your pointer hovers over an object.

You can also find object identifiers in the .mdl file. For more information, see the Transformer *Developer Guide*.

TR0514

You cannot delete or move the root category of a dimension.

The root category provides much of the information that is required by the dimension, including the label that is provided for display in the reporting components.

Do not delete a root category. If you want to remove an entire dimension from the model, select it on the dimension map, not the diagram, and click **Delete Dimension**.

TR0515

In a regular diagram, you can only connect to categories in a level above or below the original level. You cannot connect to categories in the same level.

You have tried to connect two categories in the same level.

Ensure that you only connect a category to an ancestor or descendant category. If the category that you want to connect to is at the wrong level, restructure your model so that all levels are correctly positioned in the dimensional hierarchy.

TR0518

You can only create subdimensions below categories that are in a source level.

You have tried to create a subdimension based on a category in a manual level. Transformer does not permit the creation of a subdimension below a manually-created category.

When creating a subdimension, make sure you choose a source category.

TR0519

A subdimension cannot be rooted in this level. This level is above a convergence level in a dimension that contains alternate drill-down paths.

You have tried to create a subdimension in one of the paths of an alternate drill-down structure. Subdimensions are permitted in dimensions with only one drill-down path or, in an alternate drill-down structure, at or below the convergence level.

To avoid this problem, ensure that subdimensions are positioned at or below the convergence level.

TR0523

You cannot move this category because its position is determined by its Order Value property.

You have tried to move a category manually on a diagram when the **OrderBy** value for the category has been set by one of the following methods:

- an **OrderBy** column was specified for the level in which the category resides
- an **OrderBy** value was explicitly specified for the category in the **Order Value** box on the **Category** property sheet

Ensure that you only move categories for which no **OrderBy** value is specified.

Alternatively, use one of the following methods to change the order of categories for which an **OrderBy** column is specified:

- Choose another **OrderBy** column for the level, and have Transformer re-order the categories based on this new **OrderBy** column.
- Disable the current **OrderBy** settings by choosing the blank entry at the bottom of the **SortBy Column** list on the **OrderBy** tab of the **Level** property sheet.

  Tip: You can order the categories manually by moving them on the diagram and then re-enable the **OrderBy** settings.

TR0524

Rollup can only be disabled for special categories.

The Transformer Windows interface does not allow you to clear the **Category rollup** check box when you are working with regular categories. However, this error message may appear if you try to disable rollup for a regular category using an MDL script.

To resolve the problem, ensure that the ID numbers used in the MDL script refer to objects for which the stated action is valid.

TR0525

A category was expected but not found. Either you have referred to a category by an object name or object identifier that Transformer is unable to locate, or you have not referenced the category.

This error occurs when a Model Definition Language (MDL) script requires an object and is unable to find it. In this case, the object is a category.

The problem may be that the category was not referenced, or that it was referenced with an incorrect object name or object identifier.

You can verify Transformer object names and identifiers on the Windows interface. To make them visible, select the **Object name** and **Object identifier** check boxes on the **Titles** tab of the **Preferences** property sheet. The object name and identifier will appear in the title bar at the top of each property sheet, and also in a tool tip when your pointer hovers over an object.

You can also find object identifiers in the .mdl file. For more information, see the Transformer *Developer Guide*.

TR0528

The source value for a date category in a month level must be numeric. Also, it must be in the format MMDD, unless specified otherwise in the Date tab of the **Column** property sheet.

This error can occur if the source value for a date category was inadvertently changed or deleted.

To resolve the problem, ensure that the **Source value** on the **General** tab of the **Category** property sheet uses the same date format as that specified on the **Time** tab of the **Column** property sheet.

TR0534

The drill category cannot be deleted in a dimension with a single alternate drill down.

You have tried to delete the last drill category in a dimension. Each dimension must contain at least one drill-down path.

Instead of removing the last drill-down path in the dimension, do one of the following:

- Delete the entire dimension and, if desired, recreate it with the new drill-down structure and levels you want.
- Create an alternate drill-down structure first, then delete the drill category you no longer want.

TR0535

A category cannot be created below a convergence level when the parent of that category is above the convergence level.

You have tried to create a new category below the convergence level of an alternate drill-down structure, and the new category descends from a category above the convergence level of an alternate drill-down structure. This is not a valid action.

Instead, on the diagram, create a new category at the convergence level by dragging the pointer from an ancestor level to the convergence level. Make sure you do not drag the pointer too far to the right.

Another solution is to create new categories that descend from categories in or below the convergence level.

TR0536

Partitioning can only be specified for categories in a primary drill-down path.

You have tried to specify partition numbers for categories in the alternate drill-down path of a dimension with an alternate drill-down structure. Partitioning is only supported in the primary drill-down path.

To resolve the problem, determine which categories in the alternate drill-down path have partition numbers specified, then change their partition numbers to zero (`0`).

Tip: If you are unsure which categories in the alternate-drill-down path are partitioned, an easy way to isolate categories with partition numbers other than `0` is to save the model as an .mdl file. You can then open the .mdl file using any text editor and search for the string `NewPartition`, to locate all levels or categories with partition level numbers other than `0`.

TR0538

A partition cannot be specified for root, special, or leaf categories.

You have tried to specify a partition level number for a category for which partitioning is not permitted.

To resolve the problem, ensure that you have only assigned partition level numbers to regular categories located above the lowest (leaf) level in each dimension.

Note: If a partition level number is assigned to a category in a level that is excluded, cloaked, or summarized, that category effectively becomes a leaf category in the resulting PowerCube. As a result, partition level numbers cannot be assigned to these category types. However, you can assign a partition level number to the root category of a subdimension.

TR0540

Changing the primary drill-down path from *drill_down_pathname* will cause any allocations and partitioning information to be removed from the current primary drill-down path. Do you want to continue?

Transformer supports partitioning and measure allocation only in the primary drill-down path of a dimension. If a dimension has an alternate drill-down structure, the partitioning and measure allocation cannot be in the alternate path.

If you specify partitioning or measure allocations within the primary drill-down path, and you later change the primary drill-down path to an alternate path, Transformer erases the existing partitioning and allocation information.

To fix your model, you must partition on the levels and categories in the new primary drill-down path, and also fix the measure allocation used in the new primary drill-down path.

TR0541

A drill-down category cannot be filtered.

This error occurs when you create a Model Definition Language (MDL) script that tries to exclude a drill category. The Transformer Windows interface does not allow you to exclude a drill category.

To avoid this problem, when you reference an object in MDL, make sure you use the correct object identifier or object name.

You can verify Transformer object names and identifiers on the Windows interface. To make them visible, select the **Object name** and **Object identifier** check boxes on the **Titles** tab of the **Preferences** property sheet. The object name and identifier will appear in the title bar at the top of each property sheet, and also in a tool tip when your pointer hovers over an object.

You can also find object identifiers in the .mdl file. For more information, see the Transformer *Developer Guide*.

TR0552

Manual categories in this dimension cannot be created because the dimension is locked.

You cannot create categories for a dimension that is locked.

To create a manual category for a locked dimension, clear the **Prohibit automatic creation of new categories** check box on the **General** tab of the **Dimension** property sheet.

TR0605

A column was expected but not found. Either you have referred to a column by an object name or object identifier that Transformer is unable to locate, or you have not referenced the column.

This error occurs when MDL syntax requires an object and is unable to find it. In this case, the object is a column.

The problem may be that the column was not referenced, or that it was referenced with an incorrect object name or object identifier. Or, if the column name changed, the object name may differ between MDL and the Windows interface.

MDL derives the object name from the **Original name** box on the **General** tab of the **Column** property sheet. The object name for the Windows interface is taken from the **Column name** box on the same tab.

You can verify Transformer object names and identifiers on the Windows interface. To make them visible, select the **Object name** and **Object identifier** check boxes on the **Titles** tab of the **Preferences** property sheet. The object name and identifier will appear in the title bar at the top of each property sheet, and also in a tool tip when your pointer hovers over an object.

You can also find object identifiers in the .mdl file. For more information, see the Transformer *Developer Guide*.

TR0606

The specifications for the date format and the level of detail are inconsistent. Confirm these specifications in the **Column** property sheet.

You have specified conflicting properties for the **Date input format** and the **Degree of detail** on the **Time** tab of the **Column** property sheet.

To resolve the problem, ensure that the date format provides sufficient information for the specified **Degree of detail**. For example, if you specify a **Degree of detail** of **Day**, meaning that the date values represent days, and the measures in the input records represent daily values, then you cannot specify a **Date input format** of **YM** because this does not provide sufficient detail to populate a time dimension containing days.

TR0607

The data class in the **Column** property sheet is not compatible with the storage type in the **Measure** property sheet. Change the data class or change the storage type.

The data class of a column is not compatible with the storage type of a measure that is based on that column. For example, if your database stores the Cost measure as a non-numeric `varchar`, when you try to change the data type in Transformer to `numeric`, an error occurs.

If your data source is an .iqd file and you have IBM Cognos Impromptu installed, you can resolve the problem in the source file. Change the data definition for the column using the `string-to-number` function, save the .iqd file, and then use Transformer to replace the faulty column.

For non-IQD sources, you must change the data type for the column in your source database.

TR0613

Column *column_name* is referenced by name in one or more dimensions, levels, measures or currency tables. Deleting or modifying the column name can cause these associations to be lost. Do you want to continue?

You have tried to delete a column that is currently being used in the model. Deleting a column that provides category or measure values will cause the objects in the model to lose their association with any source column.

To avoid this problem, do not delete any columns required by your model.

TR0621

An association was expected but not found. Either you have referred to an association by an object name or object identifier that Transformer is unable to locate, or you have not referenced the association.

This error occurs when an MDL script requires an object and is unable to find it. In this case, the object is an association.

The problem may be that the association was not referenced, or that it was referenced with an incorrect object name or object identifier.

You can verify Transformer object names and identifiers on the Windows interface. To make them visible, select the **Object name** and **Object identifier** check boxes on the **Titles** tab of the **Preferences** property sheet. The object name and identifier will appear in the title bar at the top of each property sheet, and also in a tool tip when your pointer hovers over an object.

You can also find object identifiers in the .mdl file. For more information, see the Transformer *Developer Guide*.

TR0623

One *association_type* association from *datasource_name* data source is already defined.

A second *association_type* association from the same data source is not allowed.

This error occurs when you try to specify a second association of the same type for the same data source.

To resolve the problem, either remove the unwanted association or make the association reference a different data source.

TR0624

One *association_type* association named *association_name* is already defined.

This error occurs

- on the Windows interface, if you try to create an association when an identical association already exists
- with MDL, when your script defines an association more than once

To resolve the problem, remove the unwanted association or make the association reference a different data source.

TR0657

The columns in data source *data_source_name* don't match the data source.

The data source has changed and one or more of the columns in the model can no longer reference its corresponding data source item.

When you create a model, the columns in each data source are saved as part of the model definition. If you reorder, add, delete, or rename the items in the data source, Transformer detects that the columns in the model no longer match those in the data source, and flags any mismatches in the Modify Columns dialog box.

For more information about the **Modify Columns** feature, see "Synchronize Columns with Your Data Source" (p. 62).

Steps for Text Data Files or IQDs

1. In the **Data Sources** list, select the query whose columns you want to modify, and from the **Tools** menu, click **Modify Columns**.
2. For data source items that do not appear as columns in the model, select the items in the **Source** list and click **Add**.
3. For columns that you want to remove from the model, select the columns in the **Model** list and click **Remove**.
4. For unmatched columns identified by a plus sign (+) in the **Matched to Source** column in the **Model** list, do one of the following:
   - To manually match query items to columns, select a data source item in the **Source** list and a column in the model, and click **Match**.
   - To allow Transformer to automatically match query items to columns, click **Auto Match**. Review any messages that appear and click **OK**.

Steps for IBM Cognos 8 Data Sources

1. In the **Data Sources** list, select the query whose columns you want to modify, and from the **Tools** menu, click **Modify Columns**.
2. If there are columns in the model that cannot be matched to data source items, you will get a warning message. Click **No** to keep these unmatched columns in the model.

   If you click **Yes**, Transformer deletes the unmatched columns.
3. For data source items that do not appear as columns in the model, select the items in the **Source** list and click **Add**.

   Tip: Click **Refresh Source** to refresh the source list for the data source. Click **Validate** to check whether columns in the model violate any Framework Manager governor settings.
4. For columns that you want to remove from the model, select the columns in the model list and click **Remove**.
5. For unmatched columns identified by and **X** in the **Matched to Source** column in the model list, do one of the following:
   - To manually match query items to columns, select a data source item in the **Source** list and a column in the model, and click **Match**.
   - To allow Transformer to automatically match query items to columns, select the columns in the model and click **Auto Match**.
6. If Transformer provides one or more locations in the data source that may be appropriate for unmatched columns, do one of the following:

- If one of the locations is an appropriate match, select the location and click **Next** or **Finish**.
- If none of the locations are an appropriate match, click **Match by reference instead** and click **Next**. From the **Candidates** list of data source items, click the one that you want to match to the column or click **Leave unmatched**.

7. Repeat step 6 for each mismatched column that Transformer attempts to resolve.
8. For any mismatched items that Transformer cannot suggest locations for, Transformer presents possible name changes for your review. From the **Candidates** list, click the item that you want to match to the column to or click **Leave unmatched**.

TR0700

This action will cause one or more cubes to be deleted in the Cube Group *cube_group_name*. Do you want to continue?

You have performed some action that will cause Transformer to delete one or more cubes from a cube group. This can happen

- if you delete a category that is in a level used to define the cube group

  This includes the deletion of categories using the **Clean House** command.

- if you create a dimension view that excludes, cloaks, or suppresses one or more of the categories used to define the cube group
- if you set the **Inclusion** setting on the **Category** property sheet to either **Suppress** or **Exclude** for a category in the level that defines the cube group
- if you use the diagram to either suppress or exclude one of the categories that defines a cube within a cube group

For example, suppose that you create a cube group based on States in a States dimension that contains levels State and Outlet. Assume that the State level contains categories State01, State02, and State03, and that Transformer creates three cubes within the cube group, one for each state. If you then create and apply a dimension view that suppresses or excludes State01, the cube for State01 will also be removed, unless you click **No** to prevent the deletion.

To avoid disrupting your cube group structure, click **No**, then fix your model design.

TR0701

A PowerCube was expected but not found. Either you have referred to a PowerCube by an object name or object identifier that Transformer is unable to locate, or you have not referenced the PowerCube.

This error occurs when MDL syntax requires an object and is unable to find it. In this case, the object is a PowerCube.

The problem may be that the PowerCube was not referenced, or that it was referenced with an incorrect object name or object identifier.

You can verify Transformer object names and identifiers on the Windows interface. To make them visible, select the **Object name** and **Object identifier** check boxes on the **Titles** tab of the **Preferences** property sheet. The object name and identifier will appear in the title bar at the top of each property sheet, and also in a tool tip when your pointer hovers over an object.

You can also find object identifiers in the .mdl file. For more information, see the Transformer *Developer Guide*.

TR0702

You cannot delete this PowerCube because it belongs to a cube group.

Transformer will not allow you to delete a cube that belongs to a cube group.

Instead of deleting the entire cube from the group, you can create a dimension view that excludes the data associated with the category you want to remove.

TR0705

Category *category_name* has been excluded from Cube Group *cube_group_name*. A PowerCube cannot be created in this cube group.

You have tried to create a cube group, but no cubes were created. The problem may be that the category or level used to create the cube group is suppressed, excluded, or cloaked.

Modify your model design to avoid creating an invalid cube group structure.

TR0713

In Cube Group *cube_group_name*, one or more cubes are based on temporary key orphanages that will be removed. Regenerate the cube group to place the data previously in the key orphanage cubes into other cubes in the cube group.

Because at least one of the cubes in the cube group was based on a key orphanage, Transformer has repaired the path from the orphaned categories to link to an existing source category in the target level for the cube group.

To resolve the problem, regenerate the cube group so that Transformer can place the orphaned data into the correct cube.

For example, assume that cube generation occurs based on data sources A, B, and C, generating the cube group in the following stages:

- Data source A is processed, and the categories in the target level are generated as part of that process.
- Data source B is processed and, during processing, a category is encountered that does not belong to any of the existing target categories. Moreover, no new data exists in data source B to provide a new target category. As a result, Transformer places the orphaned category under a key orphanage, starting at the target level, and a cube for that key orphanage is generated.
- Data source C is processed, and new data is provided that places the orphaned category under one of the existing target categories. As a result, the key orphanage is no longer required.

Because the data from data source B was originally placed in the orphaned cube, it is necessary to regenerate the cube group. This allows Transformer to place the data from data source B into the correct cube in the group.

TR0745

A total of *n* rows in the currency data were rejected because they did not match any date in the model.

The most common cause of this error is that the date loaded from an external data source is in a different format from the date format set in Transformer. For example, the date format in the external data source is **YMD** but the Transformer setting is **DMY**. It is also possible that the dates in your currency data do not correspond with the other dates in your model.

To resolve the problem, fix the date format on the **Time** tab of the **Column** property sheet. Or, in your MDL script, adjust the `Format` parameter in the Column definition statement.

TR0749

The currency table populate failed.

This error occurs when Transformer cannot find the source files needed to populate the currency table.

To resolve the problem, ensure that the data file name and path are correct or, if the currency source is an .iqd file, check the database connection string.

TR0750

Currency table Date column not found.

The **Date** column you specified does not exist, or you did not specify a **Date** column.

To resolve the problem, make sure Transformer can access the source files needed to populate the currency columns for **Date**, **Rate**, and one or both of **Label** and **Country Code**.

TR0751

Currency table Rate column not found.

The **Rate** column you specified does not exist, or you did not specify a **Rate** column.

To resolve the problem, make sure Transformer can access the source files needed to populate the currency columns for **Date**, **Rate**, and one or both of **Label** and **Country Code**.

TR0752

The currency table Label or Country Code column is mandatory.

The **Label** or **Country Code** column you specified does not exist, or you did not specify at least one of them.

To resolve the problem, make sure Transformer can access the source files needed to populate the currency columns for **Date**, **Rate**, and one or both of **Label** and **Country Code**.

TR0753

The conversion rate entered is invalid. It either contains invalid characters or is less than or equal to zero.

The source file that supplies values for the **Rate** column contains invalid data.

To resolve the problem, ensure that all conversion rates are positive numbers. The data cannot contain negative, zero, or non-numeric characters.

TR0800

Level *level_name* has no date function. Transformer will not generate any date categories if they follow a non-date level.

A non-date level is positioned in the time dimension in such a way that Transformer is unable to generate categories for any lower levels that may have date functions specified.

During category generation, Transformer proceeds down the levels of the time dimension, as indicated by the date function for each level. When Transformer encounters a level without dates, generation stops and no date categories are generated in the levels below.

To avoid this problem, make sure you do not drag a column from the **Data Sources** list and inadvertently create a new level in the time dimension.

To resolve the problem, specify a date function for the level on the **Time** tab of the **Level** property sheet.

TR0802

Levels in this time dimension are not in their natural order (year, quarter, month, day). Please reorder the levels.

An operation such as the following has been performed on date levels in the time dimension:

- The date functions for one or more date levels were changed so that the levels are no longer in the proper order.
- The date levels in the time dimension have been reordered manually.
- In an MDL script, a date level has been incorrectly referenced.

To resolve the problem on the Windows interface, reorder the date levels so that they follow the mandatory pattern for a time dimension: year, quarter, month, week, and day. For an MDL script, ensure that you reference the object using the correct object identifier or object name.

You can verify Transformer object names and identifiers on the Windows interface. To make them visible, select the **Object name** and **Object identifier** check boxes on the **Titles** tab of the **Preferences** property sheet. The object name and identifier will appear in the title bar at the top of each property sheet, and also in a tool tip when your pointer hovers over an object.

You can also find object identifiers in the .mdl file. For more information, see the Transformer *Developer Guide*.

TR0803

More than 1000 date categories have been generated from your specifications. Do you want to continue generating date categories?

You have asked Transformer to generate date categories in a time dimension. After one thousand categories, you are prompted with this warning, to give you the opportunity to stop category generation and free up system resources for other types of processing.

You can ignore this warning message if your model contains a large number of date categories; for example, if you have a custom time period with a leaf level that includes minutes or seconds. However, if you suspect that you have a runaway process, discontinue category generation and verify that your time dimension is properly defined by checking the **Time** tab settings on the **Dimension**, **Drill Category**, and **Level** property sheets.

TR0804

The earliest date is invalid.

You have specified an invalid **Earliest date** on the **Time** tab of the **Dimension** property sheet.

Verify that the dates you specify are in the valid range: between **19000101** and **99991231** inclusive. Also, the input format for earliest and latest date values must be **YYYYMMDD**.

TR0805

The latest date is invalid.

You have specified an invalid **Latest date** on the **Time** tab of the **Dimension** property sheet.

Verify that the dates you specify are in the valid range: between **19000101** and **99991231** inclusive. Also, the input format for earliest and latest date values must be **YYYYMMDD**.

TR0806

The date is not valid.

You have specified a date that is not valid, or specified a date prior to 1990 when using the **Clean House** command.

Ensure that the date is later than 1989, either by typing it in the format configured for your system, or by selecting a valid date from the embedded calendar.

TR0807

The earliest date is larger than the latest date. Make the earliest date smaller, or make the latest date larger.

You have entered an invalid combination of **Earliest date** and **Latest date** on the **Time** tab of the **Dimension** property sheet.

Ensure that the **Latest date** value is later that the **Earliest date** value.

TR0808

Invalid date format.

You have tried to specify an invalid date display format on the **Time** tab of the **Dimension** property sheet.

Ensure that the date format you have specified is a valid one.

TR0809

The current period cannot be a special category.

You have tried to set the current time period to the value represented by a special category.

Use a regular category rather than a special category to set the current period.

TR0810

The date levels specified for this dimension are not allowed. Ensure that lunar and calendar date level functions have not been combined.

You have combined both calendar and lunar levels within the same drill-down path of a dimension that contains an alternate drill-down structure.

When specifying date level functions in a time dimension, lunar time periods and calendar time periods cannot be combined within a single drill-down path. For example, it is invalid to have lunar years and calendar months in the same drill-down path.

To resolve the problem, structure your time dimension so that date level functions are specified correctly. Ensure that within a single drill-down path, all date functions are exclusively lunar periods or calendar periods.

In addition, if you have a time period in which alternate drill downs of type **Calendar** and **Lunar** converge, ensure that the convergence level is **Day** and not **Year**, **Quarter**, **Month**, or **Week**.

TR0811

The date specified by the year's start-day does not match the weekday specified by the week's start-day. Both dates must fall on the same day of the week.

You have created a lunar drill-down path in which the year-begin date does not match the day on which the week begins. Because lunar years contain 52 weeks, the date on which the year begins must match the day designated as your weekly start-day.

For a lunar time dimension, on the **Time** tab of the **Drill Category** property sheet, verify that the **Year begins** and **Week begins on** properties coincide.

For example, for a drill-down path that uses lunar time periods, if you set the **Year begins** property to 20070101, ensure that the **Week begins on** property is set to Monday, because January 01, 2007 occurred on a Monday.

TR0812

level_name is not a valid convergence level in this time dimension.

The drill-down paths leading to the convergence level do not share the same set of date categories, for one of the following reasons:

- Lunar and calendar drill-down paths are combined.
- Existing drill specifications make this convergence level illegal.

Review the documentation about creating convergence levels in the time dimension and make the necessary adjustments to your model.

TR0813

The current period for dimension *dimension_name* was not changed because category *category_code* was not found.

You have tried to change the current date from the command line using the `-t` option. Transformer was unable to find the category code specified on the command line.

Verify that you specified the correct category code for the category you want to use to set the current date.

TR0815

The interval between the year start-days for alternate drill-down paths *drill_down_name1* and *drill_down_name2* in dimension *dimension_name* must be equal to one or more whole units of time as represented by the convergence level.

You have created an alternate drill-down structure in the time dimension and specified incompatible dates for the **Year begin** days in each drill-down path.

Correct the settings on the **Time** tab of the appropriate property sheet, so the dates in each drill-down path align correctly.

TR0816

A signon was expected but not found. Either you have referred to a signon by an object name or object identifier that Transformer is unable to locate, or you have not referenced the signon.

This error occurs when MDL syntax requires an object and is unable to find it. In this case, the object is a signon.

The problem may be that the signon was not referenced, or that it was referenced with an incorrect object name or object identifier.

You can verify Transformer object names and identifiers on the Windows interface. To make them visible, select the **Object name** and **Object identifier** check boxes on the **Titles** tab of the **Preferences** property sheet. The object name and identifier will appear in the title bar at the top of each property sheet, and also in a tool tip when your pointer hovers over an object.

You can also find object identifiers in the .mdl file. For more information, see the Transformer *Developer Guide*.

TR0817

The signon may not be deleted or have its name changed because it is associated with a data source.

You have tried to remove or change the name of the signon in Transformer, which is an illegal action.

To resolve the problem, do not remove or change the name of the signon in the database connection definition. Transformer must use this name when connecting to the database.

TR0900

There were changes to the (default) earliest date and latest date. This will cause date categories to be deleted from this dimension.

You have made changes to the range of dates Transformer uses to generate date categories.

Ensure that you do not inadvertently change these default values. If the changes are intentional and you proceed, all the categories in your time dimension will be deleted.

TR0904

The date level settings could generate in excess of *n* categories. Do you want to continue?

Transformer warns you that it is about to generate an excessively large number of date categories. This can happen when you have specified a relatively wide range of dates to include in the generation of date categories and the time dimension includes low levels, such as **Day**.

In cases where your model requires all these dates, you can ignore this message. However, if you only need two or three years of data, narrow the range in the **Time** tab of the **Dimension** property sheet before you generate the categories for your model.

TR0906

The associated column that is specified for dimension *dimension_name* cannot be found in the Data Sources list.

Transformer is unable to find a column to match the one specified as the column associated with a regular time dimension. This can happen when

- the source file for a data source has changed
- you delete or change the name of a source column
- you change the column name specified on the **Time** tab of the Time Dimension property sheet

Ensure that the **Column** name specified on the **Time** tab of the **Dimension** property sheet matches a column in one or more of your data sources.

TR0907

A dimension was expected but not found. Either you have referred to a dimension by an object name or object identifier that Transformer is unable to locate, or you have not referenced the dimension.

This error occurs when MDL syntax requires an object and is unable to find it. In this case, the object is a dimension.

The problem may be that the dimension was not referenced, or that it was referenced with an incorrect object name or object identifier.

You can verify Transformer object names and identifiers on the Windows interface. To make them visible, select the **Object name** and **Object identifier** check boxes on the **Titles** tab of the **Preferences** property sheet. The object name and identifier will appear in the title bar at the top of each property sheet, and also in a tool tip when your pointer hovers over an object.

You can also find object identifiers in the .mdl file. For more information, see the Transformer *Developer Guide*.

TR0914

This dimension contains an alternate drill-down path with a convergence level. All drill-down paths that share this convergence level must use the same calendar type. A mixture of calendar and lunar date functions is not supported.

In a time dimension, you have tried to set the convergence level for two alternate drill-down paths of type **Calendar** and **Lunar** to some level other than **Day**.

Adjust your model so that the calendar and lunar year drill-down paths converge as required, at the **Day** level.

TR0917

Please specify a column for a regular time dimension.

You have tried to create a **Regular** time dimension without providing the name of a column that provides date values for the categories in the dimension.

On the **General** tab of the **Dimension** property sheet, specify the name of a data source column that provides date values for the dimension.

TR0919

Measure *measure_name* is an allocated measure in dimension *dimension_name*. Cube group *cube_group_name* is based on this dimension. This measure will not appear in the PowerCube (.mdc file) unless an external detail level is specified for the cube group.

An allocated measure was used in a PowerCube (.mdc file) that is part of a cube group, but the cube group was not defined with an external level of detail.

Specify an external level of detail, to ensure that measure values required to perform the allocation are available in the PowerCube.

TR1003

The Impromptu Query Definition contains a syntax error.

You have tried to use an.iqd file as a data source, but the contents of the .iqd file have become corrupted.

Verify that you are using the correct .iqd file. If you have IBM Cognos Impromptu installed, try to recreate the file, or investigate using Framework Manager to resolve the problem.

TR1005

IQD Data Source file *file_name* contains one or more BLOB data columns. BLOB data columns cannot be handled by Transformer.

Transformer does not support the Binary Large Object (BLOB) data type.

If you have IBM Cognos Impromptu installed, try to recreate the .iqd file, omitting the column or columns with the BLOB data, or use Framework Manager to resolve the problem.

TR1100

The import file *file_name* has an invalid keyword on line *n*.

There is an invalid keyword in the MDL script.

Check that all the keywords are spelled correctly on the specified line. For more information about supported keywords, see the Transformer *Developer Guide*.

TR1101

Transformer detected a syntax error at line *n* in file *file_name*.

There is an error in the MDL. This message is generally followed by one or more additional error messages, which may be further explained in the online help, organized by TR number.

To resolve the problem, first check the specified line, looking for obviously erroneous, invalid, or corrupt data. Fix or comment out (delete) the problem line and repeat the action to see if the process completes successfully. Alternatively, open the model on the Windows interface and run **Check Model**.

For information about the MDL syntax to use, see the Transformer *Developer Guide*.

TR1102

Transformer detected a model error at line *n* in file *file_name*.

There are several conditions that could cause this error. However, this error message is generally followed by one or more additional messages. On the Windows interface, you can search the index for information about these error messages, organized by TR number.

For information about the MDL syntax to use, see the Transformer *Developer Guide*.

TR1106

The object for the object ID given on line *n* is the wrong type in file *file_name*.

This error is generated when an object identifier specified in the .mdl file exists, but is being applied to the wrong object type.

For example, suppose 117 is the object identifier for a level but in the .mdl file, a category is identified with the object identifier 117.

To resolve the problem, use a different object identifier. Object identifiers can be any number greater than 100 and less than 4,294,967,296.

TR1109

The import file *file_name* contains a string on line *n* which contains a new line character.

This error indicates that there is a newline character on the specified line in your source file.

First, check that there is no newline character on the specified line. Then, try saving the model as a .mdl file, open it, and repeat the action that initiated the error message.

TR1304

The associated column you specified for measure *measure_name* cannot be found in the Data Sources list.

In defining the measures for your model, you have inadvertently specified the name of a column that does not exist.

Ensure that the name of the column you specify for a source measure matches the name of a column in the **Data Sources** list.

TR1307

This measure is used as a weight by measure *measure_name*.

You have tried to delete a measure that is used as a weighting for a **Regular**, **Time state**, or **Duplicates** rollup.

Before you try to delete a measure, make sure it is not being used as a weighting factor for another measure.

TR1308

This measure is referenced by calculated measure *measure_name*. Please redefine the calculated measure.

You have tried to delete a measure that is referenced by another measure.

You cannot delete this kind of measure from a model unless you first redefine the calculation or calculated measure that uses it.

TR1309

The calculated measure cannot use Before Rollup timing because it refers to a calculated measure with an After Rollup setting.

You have tried to specify a rollup setting for a calculated measure that references another measure, but the setting you chose conflicts with that used for the referenced measure.

Redefine the timing settings so that both measures use the same type of rollup. For example, you cannot specify a regular timing of **After Rollup** for a calculated measure M1, if M1 is referenced in the expression for calculated measure M2, and the timing for M2 is **Before Rollup**.

TR1310

A measure was expected but not found. Either you have referred to a measure by an object name or object identifier that Transformer is unable to locate, or you have not referenced the measure.

This error occurs when MDL syntax requires an object and is unable to find it. In this case, the object is a measure.

The problem may be that the measure was not referenced, or that it was referenced with an incorrect object name or object identifier.

You can verify Transformer object names and identifiers on the Windows interface. To make them visible, select the **Object name** and **Object identifier** check boxes on the **Titles** tab of the **Preferences** property sheet. The object name and identifier will appear in the title bar at the top of each property sheet, and also in a tool tip when your pointer hovers over an object.

You can also find object identifiers in the .mdl file. For more information, see the Transformer *Developer Guide*.

TR1312

The type of Time State Rollup used must be the default because there are multiple time dimensions.

Your model contains more than one time dimension, and when setting the type of time-state rollup, Transformer has detected rollups other than the default.

If your model contains more than one time dimension, set each one to the default time-state rollup. If a time-state rollup other than the default is required for one of your time dimensions, use that time dimension in a separate model.

TR1319

Measure *measure_name* can be in multiple data sources, but only if the lowest levels in each regular dimension associated with those data sources are identical.

This error is generated when Transformer detects a measure that obtains source values from columns in two or more data sources, but the data sources are not associated with the same levels and dimensions.

For example, suppose you have two data sources that contain source values for the Revenue measure. The data sources are not related in any other way. Both data sources have columns that are used as source values for levels in different dimensions. Transformer cannot perform rollups correctly because it does not know how to associate measure values with specific categories.

To resolve the problem, we recommend that you only obtain measure values from multiple data sources if all of those data sources contribute to the same dimension. Then, to ensure that each measure gets properly applied, create an unambiguous relationship between the measures in each data source and the dimensions in your model, using one of the following methods:

- Reconstruct your data sources so that all measure values are obtained from a single source.
- Redesign your model so that the data sources that provide measure values contribute to the same dimension.

Tip: Use the **Show Scope** command to see which data sources contribute to a particular dimension.

TR1320

Data sources associated with measure *measure_name* include levels only in the alternate drill downs and not in the primary drill down for dimension *dimension_name*. No source values will be in the cube for this measure.

In constructing the data sources for your model, you have created a measure-containing (transactional) data source that provides only category values in alternate drill-down paths. However, for measure values to be written to the PowerCube, they must be associated with the primary drill-down path.

Check the scope of the measure by using the **Show Scope** command. Then redesign your model or source file for the measure so that the required columns provide category values for the levels in the primary drill-down path.

TR1502

A data source was expected but not found. Either you have referred to a data source by an object name or object identifier that Transformer is unable to locate, or you have not referenced the data source.

This error occurs when MDL syntax requires an object and is unable to find it. In this case, the object is a data source.

The problem may be that the data source was not referenced, or that it was referenced with an incorrect object name or object identifier.

You can verify Transformer object names and identifiers on the Windows interface. To make them visible, select the **Object name** and **Object identifier** check boxes on the **Titles** tab of the **Preferences** property sheet. The object name and identifier will appear in the title bar at the top of each property sheet, and also in a tool tip when your pointer hovers over an object.

You can also find object identifiers in the .mdl file. For more information, see the Transformer *Developer Guide*.

TR1503

Choosing the 'Maximize Data Source Access Speed' option may increase processing speed in some cases but Transformer will no longer detect multiple instances of source values in levels marked as Unique. Neither will it do unique moves.

This message warns you that choosing the **Maximize Data Access Speed** optimization option for a data source disables unique moves and uniqueness error-checking.

If you want to retain these capabilities, be sure to select the **Verify Category Uniqueness** optimization option. Transformer can then check to see if the data is unique in your data source, and enable use of the **Unique Move** feature for the data source.

TR1601

Transformer could not open the model file.

The .py? or .qy?-format file that you are attempting to open is not valid.

To resolve the problem, you can restore the model from a .mdl backup file, or upgrade to the current version of Transformer (.pyj) and migrate your models as .mdl files.

TR1700

The category *category_name* in the dimension *dimension_name* has *n* immediate descendants. This number should normally be close to 7. More levels may be appropriate in this dimension.

This message is a design recommendation and will not affect your ability to generate categories or create PowerCubes.

Categories with too many immediate descendants yield reports that are difficult to use. For example, in a time dimension with only two levels, Years and Days, each Year category has 365 immediate descendants. Navigating through the resulting reports would be difficult.

To improve OLAP performance and usability, add manual levels to create a structure that limits the number of immediate descendants at each parent level. We recommend a child:parent ratio of 10:1 or less. In the above case, add intermediate levels for Quarters and Months.

Tip: You can change the warning level by changing the default **Warn when the number of child categories exceeds** setting located on the **General** tab of the **Preferences** property sheet. The default value is `35`.

TR1703

A data input conversion or overflow error occurred at source record number *n* for measure *measure_name* in source file *file_name*.

This error may be caused by an incorrect data type for the measure.

To resolve the problem, ensure that the data type is valid for, and matches, the values stored in your data source.

TR1900

The wrong ID stamp was detected on *cube_name*. This often happens when the model is not saved after creating a PowerCube.

Transformer has detected an inconsistency between the model and the PowerCube that you are trying to update.

This can occur when you try to update an incrementally updated PowerCube using a version of that cube other than the one the model was most recently saved with. For example, if you save a model after creating a PowerCube using February data, you cannot restore the January version of the PowerCube without raising this error.

To resolve the problem, click **OK** at the prompt to proceed with the incremental update and click **Cancel** to set the PowerCube status to **Invalid**. You must then **Adopt** the cube, to reset the status to **OK**.

TR1903

Incremental update of *cube_name* cannot proceed because of the *status_name* status of an existing PowerCube.

Transformer has detected a condition that prevents it from proceeding with an incremental update to a PowerCube.

For example, if you try to incrementally update a PowerCube while the cube is being viewed in a reporting component, the update may fail and the cube status may be reset to **FAILED**. If you then attempt to recreate the cube, Transformer will not allow the operation because the cube may already contain some of the data from the most recent increment. Reprocessing that increment could result in some data being added to the cube twice.

To resolve the problem, do one of the following:

- Revert to a backup copy of the model and its PowerCubes, saved prior to processing the most recent increment, and retry. When you see a second warning indicating that the time stamps on the PowerCube and model do not match, proceed with the update, accepting the risk that you might thereby duplicate some of the cube data.

- Adopt the existing cube and rerun the incremental update, having first confirmed that, if the cube was open in another component, its data has not changed.

- Rebuild your cube using the combined data from all increments.

After you resolve the problem, you can proceed with new increments in the normal way.

TR1907

Transformer cannot gain access to database *database_name* with signon information <user ID, password>.

This warning message is generated when a database signon fails when using an IQD data source. The cube may still generate successfully despite the warning.

To resolve the problem, ensure that the **Prompt for password** box on the **General** tab of the **Signon** property sheet is not selected during model creation.

TR2000

The main dimension is required in the PowerCube Group.

In a PowerCube group, you have tried to omit the dimension containing the level that defines the group.

You cannot omit this dimension. If you want to omit a portion of the dimension, use a custom dimension view. Be aware, however, that omitting one of the categories that defines a cube in the group by using the **Exclude** or **Cloak** action will cause that cube to be deleted from the group.

TR2001

The main dimension is required in the time-based partitioned PowerCube *cube_name*. It has been omitted for custom view *view_name*.

In a time-based partitioned cube, you have tried to omit the dimension containing the level that defines the cube, for the specified custom (security) view.

You cannot omit the entire dimension. If you want to omit a portion of the dimension, create a custom dimension view.

TR2002

This dimension cannot be omitted for this custom view because it is the main dimension for time-based partitioned PowerCube *cube_name* and the custom view has been added to that cube.

For a custom (security) view in a time-based partitioned cube, you have tried to omit the dimension containing the level that defines the time-based partitioned cube.

You cannot omit this dimension. If you want to omit a portion of the dimension, create a dimension view instead.

TR2306

You have specified a label column. Please specify an associated column also.

This message suggests that you may have inadvertently deleted the associations or roles from a **Level** property sheet.

To resolve the problem, ensure that the source information is complete and correct.

TR2307

A level was expected but not found. Either you have referred to a level by an object name or object identifier that Transformer is unable to locate, or you have not referenced the level.

This error occurs when MDL syntax requires an object and is unable to find it. In this case, the object is a level.

The problem may be that the level was not referenced, or that it was referenced with an incorrect object name or object identifier.

You can verify Transformer object names and identifiers on the Windows interface. To make them visible, select the **Object name** and **Object identifier** check boxes on the **Titles** tab of the **Preferences** property sheet. The object name and identifier will appear in the title bar at the top of each property sheet, and also in a tool tip when your pointer hovers over an object.

You can also find object identifiers in the .mdl file. For more information, see the Transformer *Developer Guide*.

TR2308

A date function has been chosen but the dimension is not a time dimension.

This message appears when date functions are set for a dimension that is not a time dimension.

To specify that the dimension is a time dimension on the Windows interface, set the dimension type on the **General** tab of the **Dimension** property sheet to **Time**. In MDL, include the option `DimType Date` in the Dimension definition statement.

Alternatively, remove the date functions entirely.

Note: On the Windows interface, you may need to temporarily select the **Time** dimension type on the **General** tab of the **Dimension** property sheet so you can open the **Time** tab and delete the appropriate date function(s).

TR2312

You made level *level_name* into a convergence level that connects two or more alternate drill-down paths. Categories in convergence levels must have unique source values. Can there be two categories in this level with the same source value?

You have created a convergence level but have not designated it as unique. Uniqueness is set by selecting the **Unique** check box on the **Source** tab of the **Level** property sheet. Or in MDL, it is set by the option `UniqueCategories True` in the Level definition statement.

If you select **No** in answer to the message prompt, the setting is changed to **Unique** and the required convergence level is automatically created.

If you select **Yes**, the convergence level is not created.

TR2313

You are creating a level as a convergence level that connects two or more alternate drill-down paths. Categories in convergence levels must have unique source values. Is this level truly unique?

You have created a convergence level but have not designated it as unique. Uniqueness is set by selecting the **Unique** check box on the **Source** tab of the **Level** property sheet. Or in MDL, it is set by the option `UniqueCategories True` in the Level definition statement.

If you select **Yes** in answer to the message prompt, the setting is changed to **Unique** and the required convergence level is automatically created.

If you select **No**, the convergence level is not created.

TR2314

This is not a unique level.

This error indicates that you have a problem related to level uniqueness. This can occur when

- you disable some error-checking options so that Transformer no longer verifies that unique levels contain unique categories
- you use multiple data sources in a model and do not define levels associated with columns from multiple data sources as unique
- you create an alternate drill-down structure that requires a unique convergence level, and the uniqueness property is not set
- you specify that a level is unique and Transformer detects that it is not
- there is some other problem with the source data

To resolve the problem, examine your source file and ensure that the data is valid for your model.

TR2316

The level *level_name* cannot be deleted because it is referenced by PowerCube *cube_name*.

You have tried to delete a level that defines a PowerCube group, which is not allowed.

If you want to delete the level, you must first delete or modify the PowerCube group so that Transformer no longer uses the categories in that level to define which cubes are included in the cube group.

TR2317

The level *level_name* is designated as unique. Source value *source_value* was used in an attempt to create a category in the path *drill_path_1*. *source_value* already exists in level *level_name* in the path *drill_path_2*.

This error indicates that you have a problem related to level uniqueness. This can occur when

- you disable some error-checking options so that Transformer no longer verifies that unique levels contain unique categories
- you use multiple data sources in a model and do not define levels associated with columns from multiple data sources as unique
- you create an alternate drill-down structure that requires a unique convergence level, and the uniqueness property is not set
- you specify that a level is unique and Transformer detects that it is not
- there is some other problem with the source data

To resolve the problem, examine your source file and ensure that the data is valid for your model.

TR2318

Transformer has detected *n* attempts to create a category in more than one path. Refer to the online help for a detailed explanation of level uniqueness.

This error indicates that you have a problem related to level uniqueness. This can occur when

- you disable some error-checking options so that Transformer no longer verifies that unique levels contain unique categories
- you use multiple data sources in a model and do not define levels associated with columns from multiple data sources as unique
- you create an alternate drill-down structure that requires a unique convergence level, and the uniqueness property is not set
- you specify that a level is unique and Transformer detects that it is not
- there is some other problem with the source data

To resolve the problem, examine your source file and ensure that the data is valid for your model.

TR2319

A convergence level in an alternate drill-down path must be unique.

This error indicates that you have a problem related to level uniqueness. This can occur when

- you disable some error-checking options so that Transformer no longer verifies that unique levels contain unique categories
- you use multiple data sources in a model and do not define levels associated with columns from multiple data sources as unique
- you create an alternate drill-down structure that requires a unique convergence level, and the uniqueness property is not set
- you specify that a level is unique and Transformer detects that it is not
- there is some other problem with the source data

To resolve the problem, examine your source file and ensure that the data is valid for your model.

TR2320

You are trying to sort a date level in descending order. This will create incorrect relative-time calculations. Do you want to continue?

Transformer uses the ordered set of categories in a time dimension to perform relative time calculations for relative time categories. If you reverse the order of the categories in the time dimension by sorting them in descending order, Transformer is no longer able to perform relative time calculations correctly.

To avoid this problem, when re-ordering categories in the time dimension, decide which aspect of the model is most important: the order of categories within the time dimension, or the relative time categories that are included within the time dimension.

TR2321

A share object level must be an ancestor of a share target level.

You have tried to apply a share to one or more categories in a level when the share that is currently set is at a lower level than the categories to which you are applying the share.

To avoid this problem, when you apply a share, make sure that the share is based on an ancestor of the categories to which you are applying the share.

TR2322

A share object level must be in the same dimension as its target categories.

You have tried to apply a share to one or more categories when the share is currently set to a category in another dimension. This is not supported.

To avoid this problem, when you apply a share, make sure that the share is based on an ancestor of the categories to which you are applying the share.

TR2323

The share object ID is invalid. A share object must be an object identifier of a category or level in the same dimension.

You have tried to specify an invalid share. A share object must be

- an existing category or level
- at a higher level in the same dimension
- specified by its object identifier

You can verify Transformer object names and identifiers on the Windows interface. To make them visible, select the **Object name** and **Object identifier** check boxes on the **Titles** tab of the **Preferences** property sheet. The object name and identifier will appear in the title bar at the top of each property sheet, and also in a tool tip when your pointer hovers over an object.

You can also find object identifiers in the .mdl file. For more information, see the Transformer *Developer Guide*.

TR2324

Partitioning can only be specified for levels in the primary drill-down path.

Typically, this error is written to the log file in response to an MDL script that attempts to specify a partition number for a level in an alternate drill-down path. The Windows interface does not normally allow you to specify partition level numbers in levels other than those in the primary drill-down structure.

To resolve the problem, verify that your partitioning is specified only for levels in the primary drill-down path. If partitioning is required for levels in an alternate drill-down path, consider making that path the primary drill-down path. Note, however, that changing the primary drill-down path will remove all current partitioning information from your model.

TR2325

The Short Name column you specified for level *level_name* is not in the Data Sources list.

You have specified a column to supply **Short Names** for the categories in a level, and Transformer is unable to find the column that you specified.

To avoid this problem, ensure that the column name you specify matches one that is defined in a data source for your model.

TR2326

The Description column you specified for level *level_name* is not in the Data Sources list.

You have specified a column to supply a description for the categories in a level, and Transformer is unable to find the column that you specified.

To avoid this problem, ensure that the column name you specify matches one that is defined in a data source for your model.

TR2502

A view was expected but not found. Either you have referred to a view by an object name or object identifier that Transformer is unable to locate, or you have not referenced the view.

This error occurs when MDL syntax requires an object and is unable to find it. In this case, the object is a view.

The problem may be that the view was not referenced, or that it was referenced with an incorrect object name or object identifier.

You can verify Transformer object names and identifiers on the Windows interface. To make them visible, select the **Object name** and **Object identifier** check boxes on the **Titles** tab of the **Preferences** property sheet. The object name and identifier will appear in the title bar at the top of each property sheet, and also in a tool tip when your pointer hovers over an object.

You can also find object identifiers in the .mdl file. For more information, see the Transformer *Developer Guide*.

TR2503

Special categories cannot be summarized in dimension views.

This error typically appears in the log file in response to an action requested by an MDL script. Under normal circumstances, the Windows interface does not allow you to perform exclude, summarize, or cloak actions on special categories in a dimension view.

If you encounter this error, ensure that the category object identifier that you specified in the `CatUpdate` or `CatMake` statement does not refer to a special category.

You can find object identifiers in the .mdl file. For more information, see the Transformer *Developer Guide*.

TR2504

Special categories cannot be suppressed in dimension views.

This error typically appears in the log file in response to an action requested by an MDL script. Under normal circumstances, the Windows interface does not allow you to perform exclude, summarize, or cloak actions on special categories in a dimension view.

If you encounter this error, ensure that the category object identifier that you specified in the `CatUpdate` or `CatMake` statement does not refer to a special category.

You can find object identifiers in the .mdl file. For more information, see the Transformer *Developer Guide*.

TR2505

Special categories cannot be cloaked in dimension views.

This error typically appears in the log file in response to an action requested by an MDL script. Under normal circumstances, the Windows interface does not allow you to perform exclude, summarize, or cloak actions on special categories in a dimension view.

If you encounter this error, ensure that the category object identifier that you specified in the `CatUpdate` or `CatMake` statement does not refer to a special category.

You can find object identifiers in the .mdl file. For more information, see the Transformer *Developer Guide*.

TR2506

An apex action cannot be done with special categories.

This error typically appears in the log file in response to an action requested by an MDL script. Under normal circumstances, the Windows interface does not allow you to perform exclude, summarize, or cloak actions on special categories in a dimension view.

If you encounter this error, ensure that the category object identifier that you specified in the `CatUpdate` or `CatMake` statement does not refer to a special category.

You can find object identifiers in the .mdl file. For more information, see the Transformer *Developer Guide*.

TR2507

This dimension view cannot be deleted. It is pre-defined.

You have attempted to delete one of the following predefined dimension views:

- **All Categories**
- **Omit Dimension**

These default dimension views are created for every dimension, and are used to include or exclude entire dimensions from a PowerCube.

To avoid this problem, do not delete, reorder, or rename these predefined views.

TR2508

This view is in use. If it is removed, references to it will use All Categories. Do you want to remove it?

You have tried to delete a dimension view that is currently being applied to one or more of the cubes in the **PowerCubes** list. By deleting the view, you remove any restrictions (exclude, summarize, cloak, or other actions) that were imposed on those cubes by the view. This can be an issue if you are creating cubes for specific audiences, and the removal of the view will allow some users to view data they should not be able to access.

Only respond **Yes** to this message if you are sure that removing the view will not allow users to access data that they should not be able to see. Take special care if you have used views to create cubes aimed at specific audiences.

TR2509

The position of the pre-defined views 'All Categories' and 'Omit Dimension' cannot be changed.

You have attempted to move one of the following predefined dimension views:

- **All Categories**
- **Omit Dimension**

These default dimension views are created for every dimension, and are used to include or exclude entire dimensions from a PowerCube.

To avoid this problem, do not delete, reorder, or rename these predefined views.

TR2510

The names of the predefined views 'All Categories' and 'Omit Dimension' cannot be changed.

You have attempted to modify the name of one of the following predefined dimension views:

- **All Categories**
- **Omit Dimension**

These default dimension views are created for every dimension, and are used to include or exclude entire dimensions from a PowerCube.

To avoid this problem, do not delete, reorder, or rename these predefined views.

TR2600

An error occurred during the creation of a PowerCube.

There may be a problem with the temporary files created during the cube-building process.

Check the location of the temporary files, as specified on the **Directories** tab of the **Preferences** property sheet; for example, **Data Temporary Files** (**dir1;dir2**). If a directory is not specified, Transformer saves data temporary files in the data files location specified in IBM Cognos Configuration.

Another cause of the problem may be insufficient disk space on your system. Either free up more space for these temporary files or specify a different location for them.

If these adjustments do not fix the problem, try deleting all currently existing temporary files. They may have been corrupted.

TR2601

An error occurred while saving.

Transformer was unable to save the model.

To resolve the problem, verify that the directory in which you are saving your model is one to which you have write access. If a model file with the same name already exists, verify also that the existing .py? file you are attempting to overwrite is not set to read-only.

TR2606

Transformer has detected one or more suspended models. Show the list of suspended models?

Transformer has detected .qy? files that contain checkpoints indicating there are one or more suspended models. You can view a list of these suspended models, so you can try to recover them.

To resolve the problem, select one of the following options, as presented in the recovery dialog box:

- The **Open suspended model at last checkpoint** option opens the selected model at the last checkpoint before failure, so you can view it and continue development from that checkpoint.
- The **Open last saved model and delete the checkpoint file** option opens the selected model as it was last saved, and then deletes the corresponding .qy? file.

  Note: If the model was not previously saved, Transformer cannot open it. Instead, you are again asked if you want to use the .qy? file to open the suspended model at the last checkpoint.

TR2608

Suspended models were not found.

You have asked Transformer to view any suspended models that exist, and Transformer cannot locate any suspended models.

By default, Transformer stores suspended models with the extension .qy? in your temporary directory.

If you know that suspended models should exist because Transformer failed at some point, search your path for files with the .qy? extension.

TR2700

The columns in data source *data_source_name* don't match the original source columns.

One of the source files associated with a data source has changed so that its columns no longer match the ones in the data source object defined for the model.

When you create a model, the columns in each data source are saved as part of the model definition. If you reorder, add, delete, or rename the columns in the source file, Transformer detects that the columns in the source file no longer match those in the data source, and flags any model columns that have changed on the Windows interface.

To resolve the problem, do the following:

1. Select the affected data source in the **Data Sources** list and, from the **Tools** menu, click **Modify Columns**.

2. For each changed column, click **Match**, **Add**, or **Remove**, to make the columns in the model match the columns in the data source.

TR2701

The columns specified by the alternate data source for PowerCube *cube_name* do not match the columns in Data Source *data_source_name*.

The columns in an alternative source file specified on the **General** tab of the **PowerCube** property sheet do not match the columns in the data source defined in the model.

To resolve the problem, do the following:

1. Select the affected data source in the **Data Sources** list and, from the **Tools** menu, click **Modify Columns**.
2. For each mismatched column, click **Match**, **Add**, or **Remove** to make the columns in the alternate data source for the PowerCube match the columns in the data source.

TR2702

There isn't a data source that is associated with dimension *dimension_name*.

The specified dimension has no data sources associated with it.

To resolve the problem, ensure that

- you have included all the data sources required for each dimension in your model
- the column names for each level match the name of a column in the **Data Sources** list
- you have not created a dimension that contains only manual levels
- you have not deleted a data source required for a dimension
- you have not set the **Timing** property of the data source so that the data source is never used to generate categories for the model
- you have not imported a column with a BLOB (Binary Large Objects) data type from an IBM Cognos package, which Transformer does not support

TR2703

A column in data source *data_source_name* matches level *level_name* in dimension *dimension_name* by name. Values from this data source column cannot be related to categories in this level because insufficient context is specified in the data source.

This error appears on category generation if a data source column cannot be associated with some level in the model.

You can solve the problem in the following ways:

- If the level you want to associate with the column contains only unique category values, select the **Unique** check box on the **Source** tab of the **Level** property sheet. This allows Transformer to directly associate the values from the column with the level. Transformer can then use columns from other data sources to obtain values for the ancestor categories.

- Restructure your source files so that the data source named in the error contains all the required ancestor columns.
- For all ancestor columns, ensure that the **Data Class** box on the **General** tab of the **Column** property sheet is not set to **Ignore**.

TR2704

Dimension *dimension_name* contains an unbalanced category-tree in which not all leaf categories exist in the same level.

Transformer has encountered a condition in which not all leaf categories exist at the same level. This can occur when your model uses multiple data sources and those data sources generate categories for disjoint levels.

For example, suppose two data sources are used to generate categories for a Products dimension that contains the levels Product Line, Product Brand, and Item. If one data source contains values that generate categories down to the Item level, and the other data source contains values that generate categories to only the Product Brand level, then the dimension is unbalanced: leaf categories occur at different levels in the dimension hierarchy.

To avoid this problem, ensure that all data sources generate categories to the same level in the dimension. Also, ensure that the timing for each data source in the **Data Sources** list is set so that categories are generated using the appropriate data sources.

TR2705

PowerCube *cube_name* does not contain any measures.

You have designed a Powercube that excludes all the measures referenced in the model.

To avoid this problem, on the property sheet for the PowerCube you are creating, click the **Measures** tab and ensure that you have included at least one measure.

TR2706

Level *level_name* can't be reached in drill-down path *drill_category_name* for data source *data_source_name*. A level must either be the first source level, be directly below a source level, or be designated as a unique level. All measures in data source *data_source_name* will have zero values in the PowerCube.

The level that is identified in the error message is not accessible in the listed drill-down path.

To avoid this problem, ensure that all levels associated with columns in a data source that do not contain their ancestor levels are unique. Also, ensure that any identically-named columns exist in one or more other data sources, and that these other data sources also include the columns that generate categories for the ancestor levels.

TR2707

Measure *measure_name1* is referenced by the calculated measure *measure_name2*. These measures must have the same rollup timing.

You have specified calculated measures, one of which references the other, and which have conflicting rollup timings.

For example, suppose you create a calculated measure M1 with a **Regular Timing** of **Before Rollup**. You then create calculated measure M2 with a **Regular Timing** of **After Rollup**, and you reference M1 when defining M2. The rollup timings conflict.

To avoid this problem, ensure that the rollup timings specified for cross-referenced calculated measures do not conflict.

TR2708

Regular measures *measure_name1* and *measure_name2* are referenced by the calculated measure *measure_name3*, which has a rollup timing of 'Before Rollup'. Therefore, both regular measures must be associated with the same data sources.

You have referenced two or more measures in a calculated measure that has a rollup timing of **Before Rollup**, but at least one component of the calculated measure is not available to provide the values for the calculation because the measures come from different data sources.

When referencing two or more measures in a calculated measure that has a rollup timing of **Before Rollup**, ensure that the source data for your model is structured so that all measures used in the **Before Rollup** calculated measure are contained in the same data file.

TR2710

Level *level_name* has an allocation specified for measure *measure_name1* using measure *measure_name2*. The referenced measure must include a lower level in its measure scope than the measure being allocated.

A measure that you have used to allocate another measure does not cover the portion of the dimension map over which you are attempting to allocate values.

For example, you have tried to proportionally allocate values for measure A to lower levels in a dimension using measure B. However, the domain for measure B does not include the levels in which you are attempting to allocate measures.

To avoid this problem, ensure that your allocation is based on a measure at the appropriate level in the dimensional hierarchy.

Tip: You can use the **Show Scope** command to check the scope of an allocated measure.

TR2711

There are no data sources set for category generation.

You have tried to either generate categories or create PowerCubes when no data sources are set to be run as part of the process.

To resolve the problem on the Windows interface, open the property sheet for each of the data sources in your model. On the **General** tab, ensure that the **Generate Categories** check box is selected for each data source that provides category values for the model.

TR2712

There are no data sources set for PowerCube creation.

You have tried to create a PowerCube using either the **Create PowerCubes** command or the **PowerBar** button. However, no data sources are set to be run as part of the cube creation process.

To resolve the problem on the Windows interface, open the property sheet for each of the data sources in your model. On the **General** tab, ensure that the **PowerCube creation** check box is selected for each data source that provides measure values for the cubes defined in the model.

TR2713

A circular allocation reference has been detected for measure *measure_name1* in level *level_name*. Measure *measure_name1* is allocated using another measure. At the same time another measure is allocated using *measure_name1*.

This error occurs because you have inadvertently specified an invalid (circular) allocation. Measures cannot be allocated by their own values, or by other measures that are allocated by those values.

For example, if you allocate Measure A using values for Measure B, you cannot then allocate another measure (Measure C) using values for Measure B if Measure C is already allocated using Measure A values.

To resolve the problem, specify a valid measure allocation regime that does not include a circular reference.

TR2714

A circular allocation reference has been detected for measure *measure_name1* in category *category_name*. Measure *measure_name1* is allocated using another measure. At the same time another measure is allocated using *measure_name1*.

This error occurs because you have inadvertently specified an invalid (circular) allocation. Measures cannot be allocated by their own values, or by other measures that are allocated by those values.

For example, if you allocate Measure A using values for Measure B, you cannot then allocate another measure (Measure C) using values for Measure B if Measure C is already allocated using Measure A values.

To resolve the problem, specify a valid measure allocation regime that does not include a circular reference.

TR2715

Measure *measure_name1* is allocated by the calculated measure *measure_name2*. Measure *measure_name1* is a component of the calculated measure.

This error occurs because you have inadvertently specified an invalid (circular) allocation. A calculated measure cannot reference a measure that uses that calculated measure for allocation.

For example, suppose you use the measure Fixed Costs to calculate the Fixed Cost Ratio measure, as follows:

```
Fixed Cost Ratio = (Fixed Costs/Sales) * 100
```

You cannot then use the measure Fixed Costs to allocate values for the derived measure Fixed Cost Ratio.

To resolve the problem, specify a valid measure allocation regime that does not include a circular reference.

TR2716

Category *category_name* is inaccessible in dimension *dimension_name*. No data can be reported against it in the PowerCube. A category is accessible if it or a descendant exists in a key level, or it has an ancestor with allocated measure values.

This message warns you that the dimension you have created contains one or more categories that cannot be accessed. This situation can occur for one of the following reasons:

- There is no **Current Period** category.
- You have created a special category that is not connected to a source category.

To resolve the problem, from the **Diagram** menu, select a date category and use the **Set Current Period** command. Also, ensure that the **Automatically set the current period** check box is enabled on the **Time** tab of the property sheet for the time dimension.

TR2717

Category *category_name* has no children in dimension *dimension_name* in at least one dimension view or Cube Group. This category cannot have a partition specified.

The **Check Model** tool has detected a condition in which a dimension view has caused one or more categories to become the lowest-level active categories in a dimension. These categories become leaf level categories. Transformer does not allow partitioning on leaf level categories.

To resolve the problem, you can do one of the following:

- completely repartition the model, to avoid partitioning on the categories that become leaf level categories in the dimension view
- change the dimension view so that the partitioned categories are not at the lowest level in the view

TR2718

Dimension *dimension_name* has no measures associated with it. During PowerCube generation, no data source will be able to provide data for this dimension.

In a model that reads data from multiple data sources, the data sources that are used to provide category values for the named dimension have no measures associated with them.

A possible solution is to ensure that when you set up multiple data source models, you include data sources that contain measures for each dimension. For multiple data source models, this requires you to set up

- a structural data source that provides category values for the dimension
- one or more transactional data sources that contain one or more columns with measure values
- at least one column associated with a unique level in the dimension

TR2719

Data source *data_source_name* contains no measures. It is referred to as a structural data source and should come before all measure data sources in the Data Sources list.

Transformer assumes that data sources without measures are intended to provide structure for the model. It processes these sources to generate the dimensional hierarchy before processing the sources that contain measures.

To resolve the problem, ensure that the items in the **Data Sources** list use the same sort order that Transformer uses when processing the data. Make the required adjustments to place your structural data sources at the top of the list, followed by the transactional sources.

TR2720

Measure *measure_name* has a time-state rollup specified. For this reason, no categories may be summarized or cloaked in time dimension views. Category *category_name* is summarized or cloaked in at least one view in dimension *dimension_name*.

Transformer does not allow dimension views that use **Summarize** or **Cloak** for time dimensions that specify a time-state rollup for one or more measures.

To resolve the problem, do one of the following:

- Eliminate the summarized or cloaked view from the time dimension.
- Eliminate the time-state rollup for the measure.

TR2723

A gap has been detected between partition *n* and *n+1*. No gaps in partition numbers are permitted.

You have tried to specify partition level numbers for a model, and you have inadvertently missed a number. You cannot leave a gap in the series of numbers assigned to partition levels.

For example, suppose your dimension contains four levels: 1 through 4. You cannot assign partition level 1 to the highest-level categories and partition level 3 to their immediate descendants.

In the above example, you must assign partition level 2 to categories at the level between levels 1 and 3.The same constraint applies when you specify partition level numbers across different dimensions. You must avoid setting level 1 for categories in one dimension and level 3 for categories in another dimension, without setting level 2 somewhere else.

TR2724

Partition *n* is defined in dimensions *dimension_name1* and *dimension_name2*. A partition number cannot be assigned in more than one dimension.

You have tried to specify partition level numbers for a model, and you have inadvertently assigned the same number to categories in different dimensions.

To resolve the problem, change the partition numbers currently specified for the categories in your model to avoid duplicate partition numbers in different dimensions.

TR2725

Category *category_name1* is a descendant of category *category_name2* and shares the same partition number in dimension *dimension_name*. A partition number can only occur once in a category path of a dimension.

You have tried to specify partition level numbers for a model, and you have inadvertently used the same level number more than once in the path from the root to the leaf category of a dimension.

For example, in a time dimension, you assign partition level 1 to the years 2000 through 2005, and then use the same level number for the quarters in 2006. This is valid. However, having assigned level 1 to a year, you cannot then assign the same level number to any category in any lower level of that year.

To avoid this problem, when you specify partition level numbers, ensure that you do not use the same level number more than once in the path from the root to the leaf category of a dimension.

TR2726

Category *category_name1* is a descendant of category *category_name2* in dimension *dimension_name*. The partition number specified for *category_name2* cannot be greater than the partition number of *category_name1*.

You have tried to specify partition level numbers for a model, and have inadvertently assigned numbers for the lower levels that are less than the numbers specified for their ancestors.

For example, in a time dimension, you assigned partition number 1 to categories in the Year level and number 2 to categories in the Month level. You cannot assign partition level 3 to categories in the Quarters level.

To avoid this problem, when you set up partition levels, ensure that you start numbering at 1 for the highest levels, 2 for child categories of that level, 3 for their descendant child categories, and so on.

TR2727

Measure *measure_name1* is an allocated measure which is referenced by the Before Rollup calculated measure *measure_name2*. A Before Rollup calculated measure cannot have allocated measures as components.

When setting up a calculated measure with a rollup timing of **Before Rollup**, you have tried to use an allocated measure in the expression for the calculated measure.

Because allocation is performed at run time, the allocated value is not available to perform the calculation before rollups are performed.

To avoid this problem, do not use an allocated measure in the expression for a calculated measure.

TR2728

Data Source *data_source_name* is an .iqd definition which references database *database_name*. This database has no database type specified.

Transformer is unable to locate a database type for an .iqd file, such as `OR` for Oracle or `CT` for Sybase Client Library.

Check the entry for the database in IBM Cognos Configuration or the [PowerPlay Server List] section of the .ini file. It must match the database definition specified in IBM Cognos Impromptu when the .iqd file was created.

TR2729

Measure *measure_name* has a time state rollup specified. This measure does not touch the lowest level in dimension *dimension_name* causing time state rollup values to be zero in any cubes that reference this measure.

The column providing date values for the time dimension does not contain enough detail to allow calculation of the time-state rollups for the measures in the model.

For example, you define a time-state measure called Inventory Level and your time dimension contains Year, Month, and Week. However, you discover that the dates in your source files do not provide details down to the Week level in the time dimension.

When defining time-state measures, ensure that the column providing date values for your model is sufficiently detailed to calculate the specified time-state rollups. If necessary, open the property sheet for the column associated with your time dimension and reset the **Degree of detail** property on the **Time** tab.

TR2731

Cube group *cube_group_name* is not defined in the primary drill-down path in dimension *dimension_name*. A cube group must be specified in the primary drill down of a dimension that has alternate drill downs.

You have tried to define cube groups based on a target level that exists in an alternate drill-down path. Cube groups can exist only for levels in the primary drill-down path.

To resolve the problem, first ensure that the current primary drill-down path does not contain features such as allocated measures or partitioning that are not allowed in alternate drill-down paths. Then, on the **General** tab of the appropriate **Drill Category** property sheet, make the drill-down path, in which you define the cube group, the primary drill-down path. Then convert the path that was originally the primary path to an alternative drill-down path, if required.

TR2732

The target category *category_name* in cube group *cube_group_name* which summarizes external categories cannot be suppressed or cloaked in any dimension views in dimension *dimension_name*.

In defining a cube group, you have tried to omit one of the categories in the target level by applying a dimension view that uses **Suppress** or **Cloak**.

For example, suppose you create a cube group based on the level State in the Regions dimension. The level contains three categories: California, New York, and Massachusetts. You cannot then apply a dimension view that suppresses or cloaks any of those three states.

To resolve the problem, redefine either your cube group or your dimension view so that categories are not suppressed or cloaked.

TR2733

Data Source *data_source_name* is not associated with any dimensions that are included in cube *cube_name*. This data source will be ignored during PowerCube (.mdc file) generation.

The PowerCube does not contain any dimensions associated with the named data source. Because measure values are not required, Transformer ignores the data source when it creates the cube (.mdc) file.

If you want the cube to contain information from the named data source, on the **Dimensions** tab of the **PowerCube** property sheet, include at least one dimension with which the named data source is associated.

TR2734

All cubes have been disabled. PowerCube (.mdc file) generation is not possible.

All of the PowerCubes in the model are disabled. No cube (.mdc) file can be created.

To resolve the problem, open the **PowerCube Status** dialog box from the **Tools** menu and enable at least one PowerCube so that Transformer can proceed with the cube creation process.

TR2735

Measure *measure_name1* has an average regular rollup with a weighted measure *measure_name2* specified. Both measures must exist in the same data source in order for the weighted measure to be applied.

The measure used to provide the weighting for another measure is not found in the same data source. Because Transformer analyzes one data source at a time, both measures must exist in the same data source.

To avoid this problem, ensure that measures used to weight other measures exist in the same data source.

TR2736

Measure *measure_name1* has an average time-state rollup with a weighted measure *measure_name2* specified. Both measures must exist in the same data source in order for the weighted measure to be applied.

The measure used to provide the weighting for another measure is not found in the same data source. Because Transformer analyzes one data source at a time, both measures must exist in the same data source.

To avoid this problem, ensure that measures used to weight other measures exist in the same data source.

TR2737

Measure *measure_name1* has an average duplicates rollup with a weighted measure *measure_name2* specified. Both measures must exist in the same data source in order for the weighted measure to be applied.

The measure used to provide the weighting for another measure is not found in the same data source. Because Transformer analyzes one data source at a time, both measures must exist in the same data source.

To avoid this problem, ensure that measures used to weight other measures exist in the same data source.

TR2740

Measure *measure_name* has a time-state rollup specified. There is no time dimension in this model so the time-state rollup setting will have no effect.

This message may indicate that you have inadvertently set a time-state rollup instead of a regular rollup. Time-state rollups have no effect on models that do not contain a time dimension.

Confirm that the rollups set for the specified measure are defined correctly.

TR2800

Transformer could not read the connection information for database *database_name* from file COGNOS.INI.

Transformer is unable to read the database information for a database that you are accessing from an .iqd file.

Confirm that the entry for the database connection is correctly specified in the Cognos.ini file.

TR3101

Transformer could not create the directory *directory_name*.

You may have insufficient space in your **temp** directory or on the drive where the **temp** directory resides.

To resolve the problem, try the following:

- Free up some disk space.
- Check the size of, and available space for, free conventional memory and the swap file.
- Run disk scan and defragmentation utilities on the Transformer computer, to ensure that the free hard drive space is contiguous.

TR3124

The root user class identifier in the model file was not found in the namespace. Ensure that you have selected the IBM Cognos Series 7 namespace that corresponds to the model file.

You are upgrading a model containing user class views that was created using Transformer version 7.x. The current IBM Cognos 8 namespace is not the same Access Manager namespace that the IBM Cognos Series 7 model was designed with.

In IBM Cognos 8, use the IBM Cognos Series 7 namespace that you used to create the IBM Cognos Series 7 user class views.

TR3311

An error occurred during data retrieval from the database. If available, refer to the Details for more information.

You may get this error message when you try to create a PowerCube using an .iqd file that accesses an Oracle data source. If the error only occurs during periods of peak network traffic, the problem may be that the `expire_time` parameter has been changed from its default setting (zero) in the sqlnet.ora file.

If you suspect that a default setting is at fault, please contact your database administrator.

TR3519

This calculated category definition is not valid.

The calculation expression you have defined for your calculated column is invalid because it does not follow the right syntax rules.

If you have specified incorrect parameters for your function, this error message will appear. For example, the Transformer-specific **share** function requires the first parameter to be a child category of the second parameter. Reversing the order of the child and parent categories within this function causes an error in the predefined syntax.

For more information about the syntax of supported Transformer functions, see "Functions" (p. 382).

Appendix D: Reference

This section provides a description of the Transformer Windows user interface, as well as additional information about the interface, modeling concepts, and database terminology to help you create PowerCubes for use in IBM Cognos 8. This information supplements the more task-oriented topics, providing context-sensitive help that can be accessed directly from the product.

For information about creating automation scripts using Model Definition Language (MDL), see the Transformer *Developer Guide*.

The Interface of Transformer Version 8.x

To help new users, this guide generally documents steps using the menus. However, experienced users may prefer to use the drag-and-drop approach, to expedite their work.

Context-Sensitive Help

The **Help** button on dialog boxes opens the online help in HTML format. You can then scroll to the required topic.

Welcome Page

From the Welcome page, you can create a new model, or open and edit an existing model. The Welcome page also keeps a list of the last four models that you opened or created, with the most recent model listed first. If you try to open a model that has been deleted or renamed, you receive an error message.

The list of recently used models is stored in the cogtr.xml file, which is located in the *installation_location*/configuration directory.

Property Sheets

Each object defined in a model, such as a level in the dimension map, a category in the category viewer, or an item in a list, has attributes that you can confirm or change in the property sheet for that object.

To open the property sheet for a model object, you can

- select the object, then from the **Edit** menu, click **Properties**
- select the object, then right-click and select **Properties**
- double-click the object

Lists

Most objects that you work with appear in lists. You can use these lists to add, move, or delete model objects, or to quickly open the associated property sheet.

Data Sources

The **Data Sources** list shows all the data sources used to generate model categories, associate measure values, and create PowerCubes, along with their source columns.

The icons in the **Data Sources** list indicate whether the object is a regular data column or a calculated column . If the data source is an IBM Cognos package or report, the icons will appear as they did in Framework Manager or the IBM Cognos Web studio.

The IBM Cognos package or report is a container source against which queries are defined. When the data source is a query based on an IBM Cognos package or report, other information, such as the filters and attributes used in each query, may also be shown for the appropriate subordinate level.

Measures

The **Measures** list, which also appears under the dimension map, may be empty when you first create your model. For all data sources except IBM Cognos package and report data sources, you can drag columns from the **Data Sources** list to the **Measures** list to define the measures for your model.

If you attempt to add a measure that has the same name as an existing measure, you will receive an error message indicating that another measure already has the same name. Each measure in a model must have a unique name.

A different icon is used for each measure type, as follows.

Icon	Description
	A regular measure that does not have currency conversion enabled
	A calculated measure that does not have currency conversion enabled
	A measure that has currency conversion enabled
	A calculated measure that has currency conversion enabled
	A measure folder
	A calculated measure folder

For data sources other than Framework Manager, you can drag columns from the **Data Sources** list to the **Measures** list to define the measures for your model.

Other Lists

A **PowerCubes** list shows the cubes or cube groups associated with the model. A **Custom Views** list shows versions of the cube filtered for specific reporting purposes. For more information about custom views, see "Adding Security" (p. 173).

A **Signons** list shows

- all the database signons (user IDs and encrypted passwords) used to access the IQD data source for models that reference an .iqd file
- signons created for accessing IBM Cognos 8 in batch mode
- signons for the data sources used by IBM Cognos packages and reports

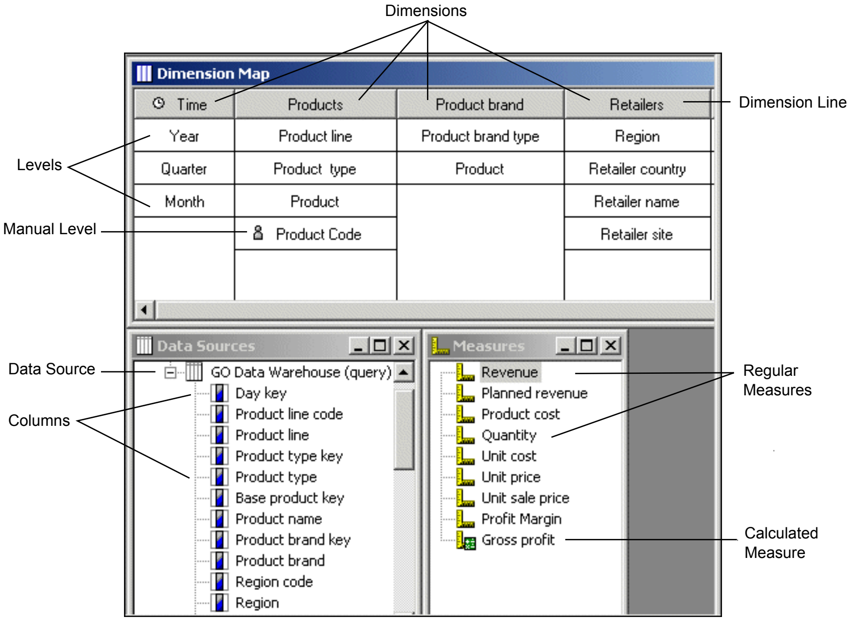
For more information, see "Adding Security" (p. 173).

Data Source Viewer and SQL Tab

The **Data Source Viewer** shows sample data from your data source. When the data source is an IQD data source or a query based on an IBM Cognos package or report, the SQL statement for your data is also shown. The **SQL** tab provides a preview of the data in Structured Query Language (SQL) form for the selected data source.

Dimension Map

The dimension map is a tabular representation of the dimensions and levels in your Transformer model. This interactive work area consists of the dimension line, which shows all the dimensions added to your model and, under that, the layered levels contained in each dimension.



Time	Products	Product brand	Retailers
Year	Product line	Product brand type	Region
Quarter	Product type	Product	Retailer country
Month	Product		Retailer name
	Product Code		Retailer site

Use the dimension map to define, check, or update the structure of your model. For example, you can

- add, move, or remove entire dimensions or the layered levels within them
- create or remove alternate drill-down paths to support multidimensional analysis
- create a time dimension that contains the relevant divisions in your calendar or fiscal year
- open the property sheet for a selected dimension or level to set its available options
- add levels manually so that you can more accurately allocate key performance measures

- lock levels or entire dimensions so that new data does not disrupt your primary structure
- show category counts for each level in every dimension
- show the scope for selected data sources or measures
- establish associations between levels and measures in a multiple data source model

The following dimension map icons are used to indicate the type of dimension or level.

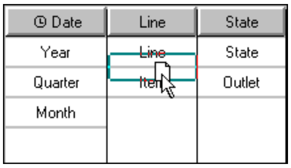
🕒	a time dimension
🔒	a locked dimension
👤	a manual level

Ellipsis points (**...**) following a level name indicates a subdimension.

When modeling on the dimension map, you can

- create a dimension map that reflects the structure of your dimensional data source
- drag a column from the **Data Sources** list to the dimension map to create a new level or dimension
- select a point in the dimension line, then add or delete dimensions from that point
- drag a level or dimension to move it to a different relative position in the dimension map

If you insert a level by dragging the corresponding column to the dimension map, a small box indicates where it will be dropped. The boundaries of the box clearly indicate whether the dropped level appears above, under, or between two existing levels, as shown.



🕒 Date	Line	State
Year	Line	State
Quarter	Item	Outlet
Month		

Scope Maps

If your model uses multiple data sources or contains measures that have meaning for some but not all the categories in each dimension, you can check the relationships between various levels in the dimension map using the **Show Scope** command on the **Edit** menu.

The possible scope states are as follows:

- Level is derived directly.

  For measures, the values are recorded in this level, or recorded in a lower level and rolled up to this level. For data sources, the level is related to a source column by name, meaning that the source records can be matched unambiguously to categories in the level. By default, this level appears with yellow shading.

- Level is derived indirectly.

For data sources, the level is not related to the source column by name but a lower level is, meaning that the source records can be matched unambiguously to categories in the level. By default, this level appears with light yellow shading.

- Level has allocated measures.

  The values for the measure are recorded in a higher level but allocated to this level. These values can be allocated by a constant or in proportion to another measure. By default, this level appears with green shading and with the same color used for each allocation type.

- Level is not derived.

  For measures, there is direct association between measure values and this level. For data sources, the level is not related to a source column by name, meaning that there is no match between records in the data source and categories in the level. By default, this level appears without any color shading (white).

- Level is derived from a data source with missing columns.

  The level is related to the source column by name but is not derived. The source records cannot be matched unambiguously to categories in the level because more than one category can have the same name in the level. By default, this level appears with red shading.

You can choose to show the scope for either a selected data source or measure. Levels are a specific color, depending on their scope and derivation. However, you can change the default color-coding for these states on the **Dimension Map** tab of the **Preferences** property sheet.

Tip: You can perform the same actions from a scope map as you can from the dimension map.

Category Counts

You can click **Show Counts** from the **Edit** menu or click the **Show Category Counts** button on the toolbar button 1001 to show

- the total number of categories in each dimension

  These appear within parentheses next to each dimension name on the dimension line.

- the number of other categories in each dimension; that is, drill categories, root categories, special categories, and categories in subdimensions

  These appear within parentheses, below the dimension names in the dimension line

- the number of regular categories in each level

  These appear within parentheses following each level name.

Diagrams

You can click **Show Diagram** from the **Diagram** menu or click the toolbar button to show

- dimensions and dimension views in a tree structure on the **Dimensions** tab (left pane)
- views based on a configured IBM Cognos namespace on the **Custom Views** tab (left pane)
- all the levels and categories in the selected dimension, in the category viewer (right pane)

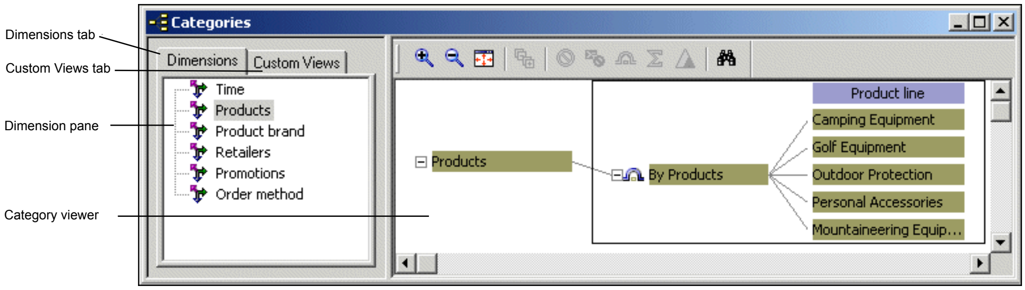
You can access the main menu, toolbar buttons, and pop-up menus from any pane and use these controls to fine-tune your model. For example, you can

- expand or collapse folders to show or hide details
- zoom in and out with the font size changing automatically to suit the degree of magnification
- access the associated property sheet by double-clicking an item
- add or delete one or more categories or levels
- use the **Show Reference** command to show the origins of a level

  From the **Tools** menu, click **Show Reference**.
- move one or more categories, provided the **Order By** option is not specified for the level
- exclude or suppress a level
- define a set of levels or categories, and apply one or more operations or calculations to them

The category viewer icons indicate which properties apply to the listed categories. The only exception is apexed categories. There is a toolbar icon △ for apexing but no viewer icon for it.

The icon ▦ for a calculation applied at the dimension level is slightly different from that used for a calculated category. We recommend that you use the **Calculation Label** option to clarify for your users the nature and scope of a calculated category maintained by a dimension.



User Interface Operations That Are Not Valid

You cannot perform the following actions:

- remove the root category from a diagram
- remove drill categories from diagrams, unless an alternate drill-down path exists
- remove measures that are used in the expression of a calculated measure
- remove cubes from a cube group
- connect categories to descendants situated at the same level

Tip: If you use the diagram to add a special or manual category with descendants, check the scope map to ensure that your additions do not conflict with other categories in the model, or cause uniqueness problems.

Connect Manual Levels Using the Diagram

After adding special or manual categories to a level, you can manually connect the descendant categories to establish the proper hierarchy.

Steps

1. In the category viewer, pause the pointer over the right side of the root category, manual category or special category to which you want to link the descendants.
2. When the pointer changes to a cross-hair, drag the pointer from the parent to the right.

   A connecting line appears dynamically to show you where a valid connection can be made.
3. Drop the pointer when it is positioned on the descendant or, for a brand new descendant category, when it is no longer positioned over any existing categories.
4. Set the properties as required for your newly connected descendant category.

Allocation Types

The following table lists the allocation settings that you can specify and provides a description of each type.

For more information about allocations, see "Allocating Measures" (p. 137).

Allocation setting	Description
From Level (**Category** property sheet only)	Uses the setting for the level in which the category resides.
Do Not Allocate	Suppresses allocation. Reports show **na** for the descendants of the category.
Constant	Allocates the measure value associated with the current category as a constant to all descendants. Reports show the constant value in all descendant categories.
By Measure	Opens the **Select a Measure** dialog box. Select a measure that provides the weighting values for the measure being allocated. For example, you can allocate the value of a fixed cost measure to various regions based on another measure, such as sales for each region.

Category Actions in Diagrams

The following are category actions that you can set in the diagram.

Exclude

The **Exclude** action eliminates categories and their descendants. The Transformer diagram shows categories and hides their descendants. Report users can drill down only to the parent of the excluded category.

You can apply the **Exclude** action at the dimension level of the category viewer, in a dimension view, and in a custom view.

Use caution when excluding categories from a dimension that contains alternate drill-down paths. If you exclude categories from one drill-down path, the data is excluded from all such paths.

Cloak

The **Cloak** action eliminates categories and their descendants, but rolls up their values for representation in the parent category. The Transformer diagram shows categories and hides their descendants. Report users can drill down only to the parent of the cloaked category.

You can apply the **Cloak** action in a dimension view and a custom view. You can remove an alternate drill-down path from a cube by cloaking the associated drill category.

You cannot cloak special categories or categories in a convergence level of an alternate drill-down path.

Suppress

The **Suppress** action eliminates categories, maintaining links from the parent to the child categories. The Transformer diagram shows categories. Report users bypass the suppressed category when they drill down.

You can apply the **Suppress** action at the dimension level of the category viewer and in a dimension view, but not in a custom view.

You cannot suppress root categories. In dimensions with alternate drill-down paths, the drill category in the primary drill-down path is suppressed by default. However, you can reverse this suppression.

Apex

Selecting the category and then clicking the **Apex** toolbar button makes this the highest-level category. The Transformer diagram suppresses all ancestor categories and hides all siblings (and the ancestors of all siblings) of the apexed category. Report users see only the apex category and its descendants.

You can apply the **Apex** action in a dimension view and a custom view.

You cannot use the apex function on a union with an alternate drill-down path and you cannot apex special categories. In an apexed dimension, any special categories that refer to regular visible categories become children of the apexed category. You can reverse this action by reapplying apex to the root category.

Summarize

The **Summarize** action eliminates descendants, but rolls up their values for representation in the selected category. The Transformer diagram shows categories. Report users can drill down only to the summarized category. They cannot see any of the category descendants.

You can apply the **Summarize** action in a dimension view and a custom view.

You cannot summarize special categories.

Category Inclusion Settings

To prevent cubes from showing zero values for categories that lack data, you can specify that categories be included in a cube only if they occur in the data source. Cubes that are more compact build faster and help report users to more quickly focus on the relevant data.

The **Inclusion** setting is available in the **Level** property sheet, **Category** property sheet, **Drill Category** property sheet and **Special Category** property sheet. If specified on a category property sheet, the **Inclusion** setting affects only the one selected category.

You can specify a default **Inclusion** setting for all categories in a level by specifying the **Inclusion** setting on the **Level** property sheet. The setting then applies to all categories subsequently created in that level.

The following table lists the available **Inclusion** settings and provides a description of each.

Tip: You may not see all the settings listed in Transformer. The settings that are available depend on the item you selected (level, category, drill category, or special category).

Inclusion setting	Description
Default (From Level)	Uses the setting for the level in which the category resides.
Always include	Retains the category in the model and includes it in cubes even if it, or any of its descendants, fails to appear in the data source.The category is also included if any of its descendants are included and if the category is not explicitly excluded, summarized, or cloaked in a dimension view. If necessary, Transformer includes ancestors of the category, regardless of their **Inclusion** settings. This is the default setting when you create a time dimension. If you exclude time categories, special categories that use relative time functions, such as **Last Month**, will not work properly.
Suppress blank categories (levels only)	Retains categories in the model, but excludes from cubes those categories with a blank source value.

Inclusion setting	Description
Suppress (categories only)	Excludes the category from the cube but retains its descendants and all data values associated with them. The immediate descendants appear in reports at the same level as the excluded category. The result is identical to using the **Suppress** option on the **Diagram** menu. For a category to be excluded, the following conditions must be true: • The category is not the share category of another. • The category is not referenced by a special category.
Include when needed	Retains the category in the model, but excludes it from cubes when it and all its descendants fail to appear in the current data source.If necessary, ancestors of the category are included, regardless of their **Inclusion** settings.
Excluded (regular and calculated categories only)	Excludes the category, its descendants, and all related data from the cube. The result is identical to using the **Exclude** option on the **Diagram** menu. For a category to be excluded, the following conditions must be true: • The category is not the share category of another. • The category is not referenced by a special category.

Date Formats and Functions

You can specify that your model use one of the supported date settings in the following table. You set date formats and functions on the **General** tab of the **Preferences** dialog box.

Date setting	Description
Predefined	As defined in one of the following supported data sources: • an Impromptu Query Definition (IQD), IBM Cognos 8 package query item, or IBM Cognos 8 report query item, where the column has a data type specified in the database • a spreadsheet with date-formatted cells • PowerHouse portable subfiles, with a column marked as date in the subfile dictionary Where the data source does not define the date format, such as in text files, Transformer assumes the format to be YYYYMMDD.
From Windows **Control Panel**	As defined in **Regional Settings**. The value is taken from the **Short Date Style** box on the **Date** tab, or its equivalent entry on your version of the Windows operating system. You can change this setting on the **General** tab of the **Preferences** property sheet, accessible from the **File** menu. When new columns are created using this setting, the **Date Input Format** shown on the **Column** property sheet mirrors the Windows **Control Panel** setting.

Date Codes

To specify a date format that exactly matches that used in your source data, on the **Time** tab of the **Column** property sheet, choose one of the following **Date Input Format** settings.

Date input format setting	Description
YMD	Year, Month, Day; for example, 070413 or 2007-Apr-13
DMY	Day, Month, Year; for example, 130407 or 13 Apr 2007
MDY	Month, Day, Year; for example, 041307 or Apr 13, 2007
YM	Year, Month: for example, 200704 or 2007-Apr. This setting is processed as if the day component were 01.
MY	Month, Year; for example, 042007. This setting is processed as if the day component were 01.
Y	Year; for example, 2007. This setting is processed as if the month and day components were 01.

To construct dates consisting of years, quarters, months, or days, use the following codes.

Date code	Description
YY	A 2-digit year; for example, 07
YYYY	A 4-digit year; for example, 2007
Q	A 1-digit quarter; for example, 1
MM	A 2-digit month; for example, 01
MMM	The abbreviated month name; for example, Jan
MMMM	The full month name; for example, January
DD	A 2-digit day; for example, 01
DDDD	A day of the week; for example, Sunday
/, -, or a space character	Alternate separator; for example, 2007/01/01 or 2007-01-01
Any quoted string	The quoted string; for example, "(" shows an open parenthesis

Tip: You can combine codes. For example, use YYYY MM DD to show dates in the format **2007 Jan 01** and use YY "Q"Q to show dates in the format **07 Q1**. For lunar years, quarters are labeled Q1-4, months are labeled 1-12 or 1-13, and days are labeled 1-28.

Date Functions

The following table describes the categories created by each supported date function setting.

Date function setting	Description
None	Use the value in the **Source** column.
Calendar year of 365 (or 366) days	Based on the standard calendar in the format YYYY or YY.
Lunar year of 52 weeks	Based on a lunar year, which contains exactly 52 weeks.
Calendar Quarter	Based on the standard calendar in the form YYYY Q or YY Q, where Q is the Quarter number 1-4.
Lunar Quarter	Based on lunar quarters, which contain exactly 13 weeks.

Date function setting	Description
Calendar month	Based on the standard calendar, in the format YYYY/MMM or YY/MMM.
4-week (lunar) month	Based on a lunar month, which contains exactly four weeks.
4-4-5 week pattern	Based on repeating 3-month patterns containing four weeks, four weeks, and five weeks.
4-5-4 week pattern	Based on repeating 3-month patterns containing four weeks, five weeks, and four weeks.
5-4-4 week pattern	Based on repeating 3-month patterns containing five weeks, four weeks, and four weeks.
Week	Based on weeks, in the format `YYYY/MMM/DD` or `YY/MMM/DD`. When using this function, you must specify how the weeks split over month boundaries, and the day that marks the start of each week, using the **Time** tab of the **Drill Category** property sheet.
Day	Use format YYYY/MMM/DD or YY/MMM/DD.

Tip: Because lunar years are one or two days shorter than calendar years, periods that use the **Lunar year of 52 weeks**, **Lunar Quarter**, **Lunar Month**, **4-4-5 week pattern**, **4-5-4 week pattern**, or **5-4-4 week pattern** date functions leave unassigned days that, over several years, comprise entire weeks. You can manage these unassigned days by selecting one of the available **Add an extra week** settings on the **Drill Category** property sheet for the appropriate drill-down path.

Default File Locations

The following tables list the default file locations used in Transformer version 8.x on Windows 2000 and Windows XP, Windows Vista, and UNIX and Linux.

Object	Default File Location (Windows 2000/ Windows XP)
Model	My Documents\Transformer\Models
Data Source	**Data files location** as specified in IBM Cognos Configuration
PowerCube	My Documents\Transformer\PowerCubes
Data Temporary File	**Temporary files location** as specified in IBM Cognos Configuration

Object	Default File Location (Windows 2000/ Windows XP)
Model Temporary File	**Temporary files location** as specified in IBM Cognos Configuration
Logs	My Documents\Transformer\Logs

Object	Default File Location (Windows Vista)
Model	Documents\Transformer\Models
Data Source	**Data files location** as specified in IBM Cognos Configuration
PowerCube	Documents\Transformer\PowerCubes
Data Temporary File	**Temporary files location** as specified in IBM Cognos Configuration
Model Temporary File	**Temporary files location** as specified in IBM Cognos Configuration
Logs	Documents\Transformer\Logs

Object	Default File Location (UNIX/ Linux)
Model	**Temporary files location** as specified in IBM Cognos Configuration
Data Source	**Data files location** as specified in IBM Cognos Configuration
PowerCube	**Temporary files location** as specified in IBM Cognos Configuration
Data Temporary File	**Temporary files location** as specified in IBM Cognos Configuration
Model Temporary File	**Temporary files location** as specified in IBM Cognos Configuration
Logs	IBM Cognos 8 logs directory. The ../logs directory is relative to the bin directory

cogtr.xml File Settings

The Transformer global preference and processing settings should be specified in the cogtr.xml file or one of the other files that are referenced by it.

The cogtr.xml file is not installed by default, but rather is created on the first use of, and exit from, the Transformer (Windows) component. It is placed in the *installation_location*/configuration directory. A sample cogtr.xml file, cogtr.xml.sample, is installed in the *installation_location*/configuration directory. This sample file shows the most common preference settings. Although you can open and manually edit this file, whenever possible, we recommend that you make your change in the Transformer **Preferences** property sheet (**File** menu).

Note: When Transformer version 8.x is installed on Windows Vista, if you do not run Transformer as an administrator and you make changes to the cogtr.xml file, the updated file is saved by default to a **Virtual Store** directory and not to the *installation_location*/configuration directory.

Global Preference Settings

You can globally change the Transformer preference settings stored in the cogtr.xml file from the command line or using the **Preferences** property sheet, accessed from the **File** menu of Transformer Windows. The cogtr.xml file is automatically checked on startup for missing or invalid entries.

If you change the default selection for any option using the **Preferences** property sheet, you must click **OK**.

AutoPartitionOff Setting

A setting of `2` ensures that model files imported into the current version of Transformer switch to use the **Auto-Partition** feature, unless one or more cubes in the model contain features that preclude use of this default optimization method. A setting of `1` was used in previous versions of Transformer to signify that **Categories** optimization was used.

MultiFileCubeThreshold Setting

You can use the `MultiFileCubeThreshold` setting in the cogtr.xml file to enable creation of multi-file PowerCubes when your cubes exceed the 2 GB threshold. To test or use the multifile feature on smaller cubes, we recommend that you set the threshold to a smaller number, such as `1000000`.

Transformer determines the number of output files needed by taking the number of data records in the cube, dividing by the threshold, and rounding up. Cube partitions are spread evenly across these multidimensional partition files (.mdp), and an additional .mdc file is added to hold the PowerCube metadata.

CenturyBreak Setting

Although this setting is not automatically created in the cogtr.xml file, you can specify a `CenturyBreak` value to determine whether two-digit years (YY) in six-digit dates are interpreted as dates in the 20th or 21st century. Transformer stores values below the cut-off date as 21st century dates.

The default setting of `CenturyBreak=20` signifies that dates 00 to 19 are interpreted as the years 2000 to 2019, and 20 to 99 were interpreted as 1920 to 1999.

You can enter any value for this setting from 0 to 99. For example, for `CenturyBreak=10`, all YY values between 00 and 09 are interpreted as 21st century dates, and all YY values at or above the cut-off date are interpreted as 20th century dates (that is, 1910 to 1999).

If your legacy data includes overlapping dates in two centuries (such as, 1900 to 1999 and 2000 to 2020), you must change your source data to use a 4-digit year format (YYYY).

MDL Save Option Setting

Use the cogtr.xml file to specify the manually maintained MDL save options.

- To set MDL save actions to use verb instead of structured format, type

  <Preference Name="VerbOutput" Value="1"/>

- To set MDL so that it does not save the object identifiers, type

  <Preference Name="ObjectIdOutput" Value="0"/>

IBM Cognos Series 7 INI File Settings Referenced by Transformer

In addition to the global preference settings, the following .ini settings are also relevant for the Series 7 IQD Bridge component that enables Transformer version 8.x to continue to support IBM Cognos Series 7 .iqd files:

- IBM Cognos Locations

  This section provides information about the location of IBM Cognos Series 7 components.

- Databases

  This section stores database definitions so that other IBM Cognos Series 7 components can use the source tables.

  For backwards compatibility, the Cognos.ini file is the default IBM Cognos Series 7 location for all database connection information.

Relative Time Settings

The following table lists the types of relative time period available for use in your models.

Relative time setting	Description
None	Not a relative time period. Use **None** to cancel a previously established relative time period without deleting the special category and its existing substructures.
Current *Period*	The current time period. For example, if the current date is 2007/05/30, then the **Current Month** is 200705.

Relative time setting	Description
Period **To-Date**	The period-to-date. For example, if the current date is 2007/05/30 • the **Quarter To-Date** category is a specially constructed version of 2006 Q2 that contains only the months of April and May (assuming a standard calendar year of four quarters beginning in January). • the **Year To-Date** category is a specially constructed version of the year 2007 that contains 2007 Q1, and a specially constructed version of Q2 that contains only the months April and May. • the **Life To-Date** category contains all previous years plus the **Year To-Date** categories.
Period **To-Date Grouped**	A group of relative time periods consisting of a period-to-date category and a period-to-date, prior period category. Here are some examples: • **Quarter To-Date Grouped** consists of a **Quarter To-Date** category and a **Quarter To-Date, Prior Quarter** category. • **Year To-Date Grouped** consists of a **Year To-Date** category and a **Year To-Date, Prior Year** category.
Last *Period*	The last period at this level. For example, if the current date is 2007/05/30 • the **Last Month** category is 2007/04 • the **Last Quarter** category is 2007 Q1, assuming a standard year of four quarters beginning in January • the **Last Year** category is 2007
Same *Period*, **Prior** *Higher-Level Period*	The same period in the most recent higher-level period. For example, if the current date is 2007/05/30, which falls in the second month of the second quarter of a standard calendar year • the **Same Month, Prior Quarter** category is 2007/02, which is the second month of the first quarter • the **Same Month, Prior Year** category is 2006/05, which is the fifth month of the previous year

Relative time setting	Description
Period **To-Date, Prior** *Period*	The periods-to-date in the context of the most recent higher-level period. For example, if the current date is 2007/05/30, which falls in the second month of the second quarter of a standard year ● the **Quarter to Date, Prior Quarter** category is a specially constructed version of 2007 Q1 that contains only the first and second months (January and February) of that quarter
Custom	A relative time period that you specify by changing the predefined category details. For more information, see "Example - Creating Custom Relative Time Periods" (p. 350).

Transformer supports the following mechanisms for setting the current period:

- automatic, based on the most recent date in the data source that updates the current period
- manual, as set on the **Time** tab of the time dimension property sheet, or on the diagram
- using the command line, by setting the `-t` option

  For more information, see "-t option" (p. 259).
- using an MDL script

  For more information, see the Transformer *Developer Guide*.

Example - Creating Custom Relative Time Periods

This example illustrates how to create a set of custom relative time periods to meet the specific reporting needs of your users.

Suppose your first period is for the month 2007/03, which is two months before the month in which the current period occurs. You specify the following details:

- Current Period = 2007/05/30
- Basic Approach = Single Category
- Target Period = Month
- Target Offset = -2
- Context Period = Year
- Context Offset = 0

You also need a time period for 2006/10, which is two quarters ago, one month before the month in which the current period occurs in the current quarter. You specify the following details:

- Current Period = 2007/05/30
- Basic Approach = Single Category

- Target Period = Month
- Target Offset = -1
- Context Period = Quarter
- Context Offset = -2

Next, you need a time period that spans all years up to 2007, as well the first quarter of 2007 and the months April and May, 2007. You specify the following details:

- Current Period = 2007/05/30
- Basic Approach = Period To-Date Total
- To-date Period = Life
- Target Period = Month
- Target Offset = 0
- Context Period = Quarter
- Context Offset = 0

You need a time period that spans the quarters 2006 Q1, 2006 Q2, 2006 Q3, and the months 2006/10 and 2006/11. This is a Year-to-Date category two quarters ago, up to the same month in that quarter as the current month is in its quarter (the second month). You specify the following details:

- Current Period = 2007/05/30
- Basic Approach = Period To-Date Total
- To-date Period = Year
- Target Period = Month
- Target Offset = 0
- Context Period = Quarter
- Context Offset = -2

Finally, you need a time period that spans a five-month period ending in July 2006. This is a five-month running total, starting three quarters ago, where the ending month in that quarter is one month before the position of the current month in the current quarter. You specify the following details:

- Current Period = 2007/05/30
- Basic Approach = N-Period Running Total
- Number of Periods = 5
- Target Period = Month
- Target Offset = -1

- Context Period = Quarter
- Context Offset = -3

PowerCube Optimization Methods

The following table lists the types of optimization methods available for use in your models.

Method	Description
Auto-Partition	Enables the **Auto-Partition** tab, where you can set the parameters for Transformer to devise a partitioning scheme. This is the default optimization setting.
Categories	Minimizes the number of categories in a cube. Transformer adds only categories that are referenced in the source data or specifically designated to be included. Categories optimization requires an extra data pass for each cube to find the categories required for that cube.
Data Passes	Optimizes the number of passes through the temporary working files during the creation of a cube. Transformer assumes that all categories are required in the resulting cube, and does not pass through the source data to determine which categories are required. Although included, unreferenced categories are not visible in your reporting component.
Direct Create	Adds all categories in the model to the cube before the data sources are processed. Records that do not generate new categories are then directly updated to the cube. This optimization method is best used with models that are expected to generate few new categories, and where all categories are expected to be added to the cube. **Note:** This setting is not available for individual cubes in a cube group.

Rollup Functions

The rollup functions specify how measure values are evaluated in the reporting components. There are three rollup types:

- **Regular Rollup**

  Measure values are summarized from lower to higher category levels. Transformer applies these functions when the cube is created. The reporting components apply them at run time.

- **Time State Rollup**

Transformer represents the state of a measure at specific times.

For example, if a model tracks the number of active customers at the end of each quarter, you can set up a time state measure to report the number of customers active at a specific time. This is more useful than the quarterly sum of the number of customers served during each month in a quarter.

- **Duplicates Rollup**

  Transformer evaluates duplicate records in the source data.

When you enable consolidation on the **General** tab of the PowerCube property sheet, Transformer performs the duplicates rollup first, followed by the regular rollup.

Regular Rollup

The following table lists the regular rollup options and explains how the results appear in the reporting components.

Regular rollup option	Results in reporting component
Sum	Shows the sum of the values of all descendant categories of the current category.
Minimum	Shows the minimum data value among all descendant categories of the current category.
Maximum	Shows the maximum data value among all descendant categories of the current category.
Average	Shows the average of the values of all records of descendant categories of the current category.
Count	Shows the number of records that contain non-null values in all descendant categories of the current category.
Count All	Shows the number of records, including those containing null values for this measure, for all descendant categories of the current category.
Any	Shows the value 1 if any records for a descendant category contain values. Shows the value 0 if no records exist for this measure or all records that do exist have null values for this measure.
External	Shows source values that were directly assigned to specific data records.

Time State Rollup

The following table lists the time state rollup options and explains how the results appear in the reporting components.

Time state rollup option	Results in reporting component
None (regular rollup)	Shows rollups not associated with a time state measure. Use when the measure is not a time state measure.
Minimum	Shows the smallest measure value from all categories in the time period being examined.
Maximum	Shows the largest measure value from all categories in the time period being examined.
First Period	Shows the measure value from the first subordinate period in the time period being examined. For example, a time dimension contains years, quarters, and months, and you are examining data at the quarter level. For each quarter, the **First Period** option reports the measure value from the first month of the quarter. When you examine data at the year level, the option reports the first value from the first month in the first quarter of each year.
Last Period	Shows the measure value from the last subordinate period in the time period being examined. For example, if a time dimension contains years, quarters, and months, and you are examining data at the quarter level, for each quarter, the **Last Period** option reports the measure value from the last month of each quarter. When you examine data at the year level, reports the value from the last month in the last quarter of each year.
Current Period	Shows the measure value from the category that is designated as the current period in the time dimension. If the time period being examined does not include the current period, the result is identical to **Last Period**. For example, a time dimension contains years, quarters, and months, and Quarter 1 starts in January. The current period is set to April 2007. At the year level, the **Current Period** option reports the measure value for April 2007. At the quarter level, the option reports the measure value for April in Quarter 2 because April is the current period, but it shows the value of the last active month in every other quarter; that is, March for Quarter 1, September for Quarter 3, and December for Quarter 4.
Average	Shows the average of the measure values from all categories in the time period being examined.

Duplicates Rollup

The following table lists the available duplicates rollup options and explains how the results appear in the reporting components.

Duplicates rollup option	Results in reporting component
None (regular rollup)	Uses the **Regular Rollup** function specified for the measure.
Sum	Sums the measure values found in the duplicate records, except for external rollups, which override this function.
Minimum	Takes the smallest of all values for the measure found in the duplicate records.
Maximum	Takes the largest of all values for the measure found in the duplicate records.
Average	Takes the average of all values for the measure found in the duplicate records. If the **Regular Rollup** option is also set to **Average**, duplicates rollup is performed before regular rollup.
First	Takes the first value of all values for the measure found in the duplicate records by using the order of the records in the data source.
Last	Takes the last value of all values for the measure found in the duplicate records by using the order of the records in the data source.

A **Duplicates Rollup** setting other than **None** forces Transformer to consolidate the source file when generating cubes, as described in the following table.

Consolidation setting	Transformer action
Default or **Yes** (with sort)	Sorts the source file, then consolidates duplicate records by using the **Duplicates Rollup** function.
No	Overrides the **Consolidation** setting, sorts the source file, and does not consolidate duplicate records. The duplicate rollup rule is ignored.
Yes (presort)	Consolidates the source data by using the **Duplicates Rollup** function without first sorting the source file.

Special Characters

Transformer and Model Definition Language (MDL) scripts use various kinds of special character.

Quotation Marks in Data

The MDL syntax reserves single and double quotation marks (`'` and `"`) to delimit textual data. All text strings are enclosed in either double or single quotation marks.

When a text string contains both single and double quotation marks, Transformer changes the quotation marks to avoid ambiguity. All quotation marks in the original text are converted to single quotation marks and the entire string is enclosed in double quotation marks.

For example, in MDL syntax, New York becomes "New York" and Other's "No-Name" Brands becomes "Other's 'No-name' Brands". Similarly, "London" and London ' ("prime") becomes "'London' and London ' ('prime')".

We do not recommend using quotation marks to designate imperial measurements, such as inches or feet. When the quotation marks are switched, your users see incorrect values. For example, `Swivel Castor 2 1/2" Black` becomes `"Swivel Castor 2 1/2' Black"`.

Search Wildcards

You can use the following wildcard characters in the **Find in Category** dialog box when the **Use Wildcards** check box is selected.

Search character	Search results	Example
^	The beginning of a string	^inter finds interesting and interfere, but not splinters.
$	The end of a string	in$ finds in and within, but not interfere.
?	Any single character except newline	to? finds top and ton, but not to.
~	Zero or one occurrence of the preceding character or sub-expression	files~ finds file and files, but not filed.
*	Zero or more occurrences of any characters except newline	can* finds can and Canada.
#	Zero or more occurrences of the preceding character or sub-expression	filex# finds file and filexxx.
@	One or more occurrences of the preceding character or sub-expression	filex@ finds filex and filexxx, but not file.
|	Either the preceding character or sub-expression, or the following one	localis|ze finds either localise or localize.

Search character	Search results	Example
[ ]	Any character within the brackets • Ranges of characters within brackets can be used by using a hyphen. For example, [a-m]xxxxxx finds any word that begins with any letter from a to m, followed by xxxxxx. • A hyphen at the start of the information is an exception to the first example above. When it is the first character in the brackets, it also searches on itself (and everything else in the brackets). In the example, Apr[-/]07 finds both Apr-07 and Apr/07. In this case, the hyphen is not used to define a range but as a character that will be searched on.	p[iu]ck finds pick and puck. [a-d]ay finds bay and day, but not may. m[!u]st finds mist but not must. m[!a-o]st finds must but not most. Apr[-/]07 finds Apr-07 or Apr/07.
( )	Sub-expressions, so that repetition and alternative wildcard characters can be applied more generally	ab(cd)#e finds ab followed by zero or more cd combinations followed by e.
\	The next character, literally The backslash (\) means that wildcard characters can be treated as normal characters.	what\? finds what?.

Reserved Characters

The following characters are reserved and cannot be used unless they are preceded by a back slash: `{m,n}`, and `\:x`.

For example, to find `{m,n}` literally, use `\{` m,n `\}`.

Transformer Samples

The Transformer version 8.x installation includes sample data sources, models, and cubes.

Before you can use the sample databases, IBM Cognos 8 must be installed, configured, and running, and the samples must be set up.

For more information, see "Setting Up the Samples" in the IBM Cognos 8 *Installation and Configuration Guide*.

Sample Models

Transformer includes the following sample models.

Model Name and Description	Data Source	Location
Sales and Marketing.mdl Transformer model based on the GO Data Warehouse query package.	GO Data Warehouse (query) in IBM Cognos Connection	*installation_location*/webcontent/samples/models/Transformer8
Employee Expenses.mdl Transformer model based on reports that use the GO Data Warehouse query package.	GO Data Warehouse (query) in IBM Cognos Connection	*installation_location*/webcontent/samples/models/Transformer8
Great Outdoors Sales.mdl Transformer model based on the Sales query package.	GO Sales (query) in IBM Cognos Connection	*installation_location*/webcontent/samples/models/Transformer8

Sample PowerCubes

Transformer includes the following sample PowerCubes.

Model Name	Model	Location
sales and marketing.mdc	Sales and Marketing.mdl	*installation_location*/webcontent/samples/models/Transformer8
employee expenses.mdc	Employee expenses.mdl	*installation_location*/webcontent/samples/models/Transformer8
go accessories.mdc	Employee expenses.mdl	*installation_location*/webcontent/samples/datasources/cubes
go americas.mdc	Employee expenses.mdl	*installation_location*/webcontent/samples/datasources/cubes
go asia pacific.mdc	Employee expenses.mdl	*installation_location*/webcontent/samples/datasources/cubes

Model Name	Model	Location
great outdoors sales.mdc	Great Outdoors Sales.mdl	*installation_location*/webcontent/samples/datasources/cubes

Sample Data Sources

Transformer includes the following sample data sources.

Data Source Name	Type	Location
Employee Expenses	IBM Cognos 8 list report	In IBM Cognos Connection, in the **GO Data Warehouse (query)**, **Report Studio Report Samples** folder
Health Insurance	IBM Cognos 8 list report	In IBM Cognos Connection, in the **GO Data Warehouse (query)**, **Report Studio Report Samples** folder
Overtime	IBM Cognos 8 list report	In IBM Cognos Connection, in the **GO Data Warehouse (query)**, **Report Studio Report Samples** folder
Pension Plan	IBM Cognos 8 list report	In IBM Cognos Connection, in the **GO Data Warehouse (query)**, **Report Studio Report Samples** folder
Regular Salary	IBM Cognos 8 list report	In IBM Cognos Connection, in the **GO Data Warehouse (query)**, **Report Studio Report Samples** folder

Appendix E: Guidelines for Working with SAP BW Data for Use in Transformer

Transformer Version 8.4

In Transformer version 8.4, you can use Framework Manager packages published to Content Manager to leverage your SAP BW data. The SAP-based packages can be used as data sources to create Transformer models. As result, Transformer PowerCubes can be used as high speed data access cache methods for distributing smaller or focused areas of your business information. This is the recommended method for leveraging your SAP BW data.

There are special considerations when using SAP-based packages created in Framework Manager. For detailed information about creating your SAP queries, creating the SAP-based packages in Framework Manager and using them in Transformer, see "Work with SAP BW Data Using a Package in Framework Manager" (p. 361). For general information about creating packages in Framework Manager, see "Create or Modify a Package" in the Framework Manager *User Guide*.

Transformer versions 7.x, 8.1, 8.2 and 8.3

In Transformer versions 7.x, 8.1, and 8.2, you can leverage your SAP BW data using a Framework Manager package in which the query subjects and dimensions are externalized using CSV files. Transformer can use the CSV files as a data source to create a model and generate PowerCubes. This method should only be used in a IBM Cognos 8 environment when you want to leverage data in IBM Cognos 8 to build PowerCubes for IBM Cognos 8.

Note: IQD files generated by Framework Manager will continue to be supported in this IBM Cognos 8 release, but will not be enhanced. Support for IQD files will remain until the end-of-life of Transformer version 7.x.

In Transformer version 8.3, use published packages for dimensions and use CSV files for facts.

There are special considerations when using externalized CSV files with SAP data in Framework Manager. For more information, see "Working with SAP BW Data Using Externalized CSV Files in Framework Manager" (p. 371). For general information about externalizing query subjects and dimensions using the CSV method, see "Externalizing Query Subjects and Dimensions" in the Framework Manager *User Guide*.

Work with SAP BW Data Using a Package in Framework Manager

You can leverage SAP BW data in Transformer version 8.4 by using an SAP-based package created in Framework Manager and published to Content Manager. This is the recommended method to leverage your SAP BW data. There are special considerations when using SAP-based packages created in Framework Manager. For a complete set of procedures about how to create an SAP BW query, create a package in Framework Manager, and create a model using the SAP package in Transformer, see "Importing Data from an SAP BW Query Source" (p. 362).

For general information about creating packages, see "Create or Modify a Package" in the Framework Manager *User Guide*.

Importing Data from an SAP BW Query Source

You can use Transformer to import both dimensional and fact data from an SAP BW query source. The following instructions describe how to rebuild an SAP BW cube as an IBM Cognos Transformer cube. To do so, the SAP BW query package must be in a specific format.

Limitations

- This extract process is limited to SAP BW data sources only.

- The data source must be a specifically constructed query defined in the SAP BW data source.

Process

There are three stages to importing a SAP BW query to access both dimensions and facts using IBM Cognos 8:

The following sections describes each of these stages:

❑ Creating a Query in SAP BW Business Explorer Query Designer

❑ Creating a Package in Framework Manager

❑ Creating a Model in Transformer

Creating a BW Query in SAP Business Explorer Query Designer

You must create a query that includes the cube that you wish to import. We recommend that you base the query on a single InfoCube in the database. A query based on multiple sources may result in SAP BW errors during data retrieval.

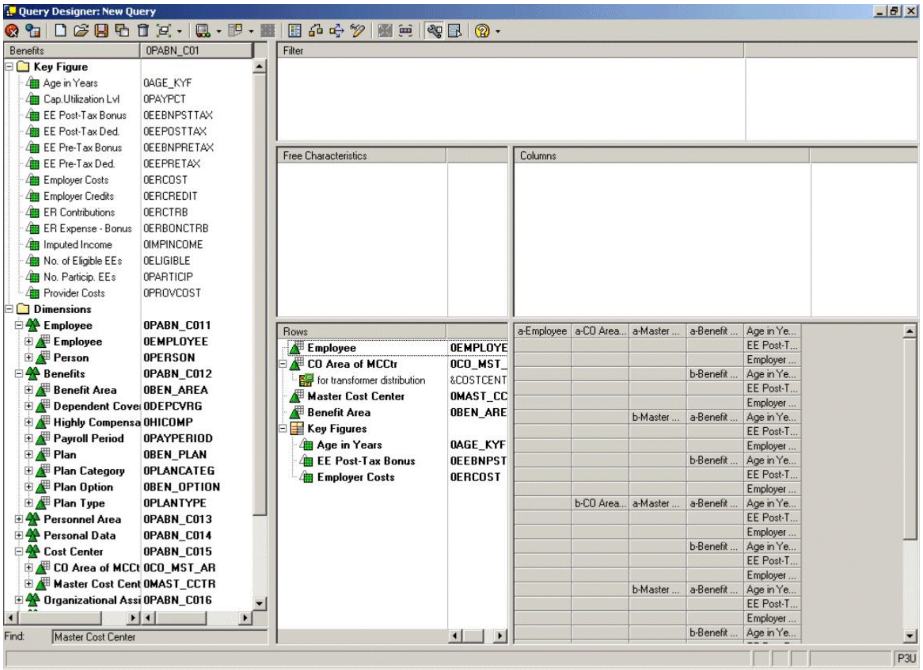
Steps

1. In **Query Designer**, click **New Query**.

2. In the **New Query** dialog box, select the information provider that contains the cube that you want to import.

3. Click the **Tools** icon to view the technical name of the **InfoObject**.

4. Drag a characteristic that you wish to import from the **InfoObject** catalog on the left column to one of the fields on the right-hand side of the page. For example, **Columns** or **Rows**.



Benefits	0PABN_C01
Key Figure	
Age in Years	0AGE_KYF
Cap.Utilization Lvl	0PAYPCT
EE Post-Tax Bonus	0EEBNPSTTAX
EE Post-Tax Ded.	0EEPOSTTAX
EE Pre-Tax Bonus	0EEBNPRETAX
EE Pre-Tax Ded.	0EEPRETAX
Employer Costs	0ERCOST
Employer Credits	0ERCREDIT
ER Contributions	0ERCTRB
ER Expense - Bonus	0ERBONCTRB
Imputed Income	0IMPINCOME
No. of Eligible EEs	0ELIGIBLE
No. Particip. EEs	0PARTICIP
Provider Costs	0PROVCOST
Dimensions	
Employee	**0PABN_C011**
Employee	**0EMPLOYEE**
Person	**0PERSON**
Benefits	**0PABN_C012**
Benefit Area	**0BEN_AREA**
Dependent Cover	**0DEPCVRG**
Highly Compensa	**0HICOMP**
Payroll Period	**0PAYPERIOD**
Plan	**0BEN_PLAN**
Plan Category	**0PLANCATEG**
Plan Option	**0BEN_OPTION**
Plan Type	**0PLANTYPE**
Personnel Area	**0PABN_C013**
Personal Data	**0PABN_C014**
Cost Center	**0PABN_C015**
CO Area of MCCt	**0CO_MST_AR**
Master Cost Cent	**0MAST_CCTR**
Organizational Assi	**0PABN_C016**

Find: Master Cost Center

Rows	
Employee	**0EMPLOYE**
CO Area of MCCtr	**0CO_MST_**
for transformer distribution	&COSTCENT
Master Cost Center	**0MAST_CC**
Benefit Area	**0BEN_ARE**
Key Figures	
Age in Years	**0AGE_KYF**
EE Post-Tax Bonus	**0EEBNPST**
Employer Costs	**0ERCOST**

a-Employee	a-CO Area...	a-Master ...	a-Benefit ...	Age in Ye...
				EE Post-T...
				Employer ...
			b-Benefit ...	Age in Ye...
				EE Post-T...
				Employer ...
		b-Master ...	a-Benefit ...	Age in Ye...
				EE Post-T...
				Employer ...
			b-Benefit ...	Age in Ye...
				EE Post-T...
				Employer ...
	b-CO Area...	a-Master ...	a-Benefit ...	Age in Ye...
				EE Post-T...
				Employer ...
			b-Benefit ...	Age in Ye...
				EE Post-T...
				Employer ...
		b-Master ...	a-Benefit ...	Age in Ye...
				EE Post-T...
				Employer ...
			b-Benefit ...	Age in Ye...

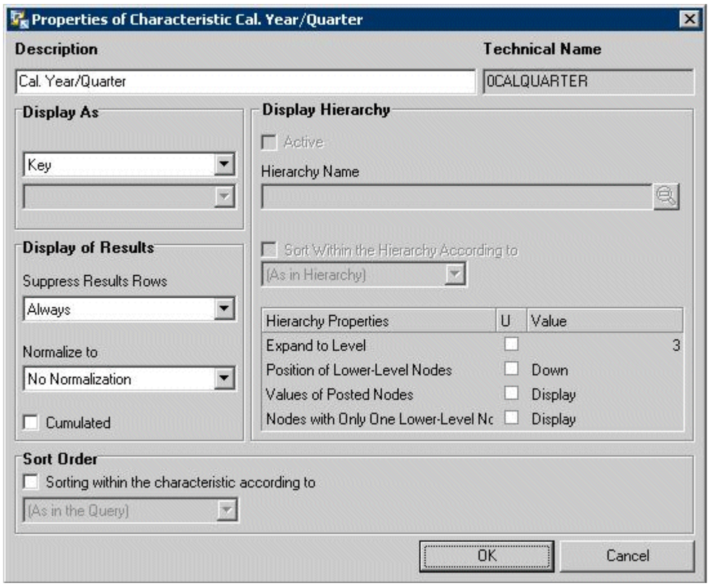
The characteristics you select will define the metadata in the Transformer cube. The characteristics must adhere to the following restrictions:

- You must have at least a single optional variable to segment the data.
- Select a characteristic that is representative of the data source. The characteristics can be either key figures, which will become measures in Transformer, or dimensions, which will become the Transformer dimensions.
- Do not assign any of the characteristics a display hierarchy, either explicitly or by a variable.
- All key figures in the SAP BW query must be numeric.
- Do not select the **Currency/Unit** characteristic.
- Ensure that all selected key figures use the same currency.
- Only include characteristics in the SAP BW query that you wish to extract using Framework Manager. Including unnecessary characteristics increases data volume, thereby adversely affecting performance.
- Characteristics must be copied to the **Dimensions** or **Rows** fields of the query definition. If copied to the **Free Characteristics** or the **Filter** fields, the characteristics show as dimensions when importing from the package but the stream extract processing is not able to fetch the values.
- If you have filters defined, they must reference only dimensions that have been included elsewhere in the query definition.
- If you include a free characteristic, no values will appear for that characteristic in the key figures extract. A filter on a free characteristic acts as a filter on the returned SAP BW data. You can use this as a filter to define a subset of an InfoCube.

- Use a picklist prompt, rather than a type-in prompt for the query. A picklist prompt provides values for segmenting the data.

5. To define the metadata that will populate the Transformer cube, you must change the properties of each characteristic that you have selected for inclusion. Right-click a characteristic, and select **Properties**.

6. In the **Properties of Characteristic** dialog box, change the **Display As** value to **Key**, and the **Suppress Results Rows** value to **Always**. Note that any restriction or filter applied here will be carried forward in Transformer.



Properties of Characteristic Cal. Year/Quarter

Description: Cal. Year/Quarter

Technical Name: 0CALQUARTER

Display As: Key

Display Hierarchy

□ Active

Hierarchy Name

□ Sort Within the Hierarchy According to

(As in Hierarchy)

Display of Results

Suppress Results Rows: Always

Normalize to: No Normalization

□ Cumulated

Hierarchy Properties	U	Value
Expand to Level	□	3
Position of Lower-Level Nodes	□	Down
Values of Posted Nodes	□	Display
Nodes with Only One Lower-Level Nc	□	Display

Sort Order

□ Sorting within the characteristic according to

(As in the Query)

7. Repeat steps 5 and 6 for each characteristic that you selected in step 4.

   Note: You should only select the characteristics that you require. To avoid excessive memory consumption, and decreased system performance or failure, carefully consider what characteristics you want to include in the query. We recommend that you consult an SAP BW administrator to ensure that the data volumes are not exceeded.

8. Click the **Queries Properties** icon, and in the **Extended** tab select the **Allow External Access to this Query** check box. This exposes the query to Framework Manager.

9. Click **Save**, and provide the new query with a **Description** and a **Technical Name**. We recommend that you use the SAP BW naming convention in the **Technical Name** field. That is, begin the entry with the letter "Z" followed by an intuitive name or your standard naming convention. It is important to write down this technical name, as you will need it to find the query in Framework Manager.

You are now ready to create a variable. For more information on using the **SAP Query Designer**, see your SAP BW documentation.

Creating a Variable

You must now create an optional prompt parameter for the query so Transformer can issue smaller Queries to SAP, and thereby retrieve the entire data set.

Guidelines for Using SAP BW Fact Data in Transformer 8.4

There are no set rules for variable usage when extracting SAP BW data for use in Transformer. However, you must be careful not to request too much data that could potentially perform poorly or error out with out-of-memory messages within your SAP environment.

A basic guideline to follow is that when a variable is utilized for the extraction, Transformer will first fetch all members that exist for the dimension against which the variable is defined. After this, Transformer will perform individual data fetches to extract the fact data for each of the individual members within the dimension in order to satisfy the variable.

This allows Transformer to break down your data extraction into manageable chunks that the SAP BW server can handle. There are no set standards as to which dimension to apply it to. To achieve optimal performance, you must understand your SAP BW data and determine which dimension evenly breaks up the factual data.

You must choose carefully which dimension to define the variable on. It may require some experimentation to achieve optimal performance. For example, you may have a [COUNTRY] dimension that contains three countries as members, United States (US), Canada (CA), and Mexico (MX). If most of the business is performed in the US (90%) and the remaining business (10%) is recorded against Canada and Mexico evenly, this dimension would not evenly split up the data. The resulting queries would have one very large request (US) and two small ones (CA and MX). Therefore, this dimension would not be a good candidate.

You do not want to apply a variable on a dimension that would cause too many very small requests. For example, [0MATERIAL], a dimension often utilized in SAP BW environments would probably not be a good candidate because it would cause too many small requests to be performed.

You may have a dimension defined for [COSTCENTER] that evenly divides up the data for 10 distinct cost centers that may serve to segment the data evenly. Another good alternative may be calendar year or calendar month because it may divide your data into sections that perform adequately.

It is not necessary to apply any variables to queries for data extraction. Some extraction will perform perfectly well when no variables are applied. For example, a good approach may be to apply a variable on a dimension which splits the data into 20 individual fetches and test the extraction. If this performs well, you may choose to apply a variable on a different dimension which may contain 5 distinct members and see how it compares.

No formula can be applied as no two environments are alike. However, a cautious approach is recommended to avoid disrupting your SAP BW environment.

Steps

1. In **Query Designer**, right-click a characteristic that you have selected in the previous procedure and select **Restrict**.

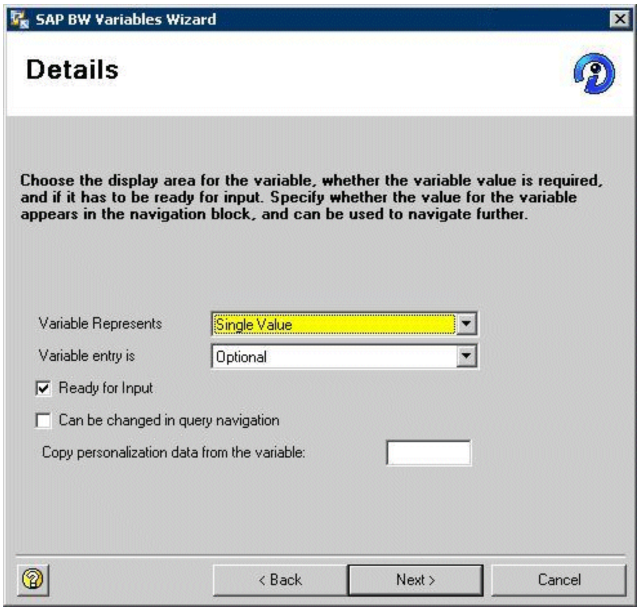
   To ensure that data is distributed evenly, select a characteristic that is representative of the cube and will not result in a large number of variables. You want a resulting variable where the number of rows for each value of the variable is similar; you do not want a resulting variable that is too fine-grained (for example, not many rows per value resulting in an excessive number of queries), nor do you want a variable that is too coarse-grained (for example, more than one million rows per value).

2. In the **Selection for ...** dialog box, click the **Variables** tab, right-click anywhere inside the **Description** window and select **New Variable**.

   Note: If one of the characteristics that you have chosen already has a variable, you can avoid creating a new variable and skip to step 7 of this procedure.

3. In the **New Variable Wizard General Information** page, type a **Variable Name** and **Description**, and select **Market** as the characteristic. Click **Next**.

4. In the **Details** page, select **Single Value**, **Multiple Single Values**, or **Interval** in the **Variable Represents** field, **Optional** in the **Variable entry is** field, and select the **Ready for Input** check box. Click **Next**.



Choose the display area for the variable, whether the variable value is required, and if it has to be ready for input. Specify whether the value for the variable appears in the navigation block, and can be used to navigate further.

Variable Represents: Single Value

Variable entry is: Optional

☑ Ready for Input

□ Can be changed in query navigation

Copy personalization data from the variable:

5. In the **Default Values** page, ensure that the **Default Value** field is empty.

6. Click **Next** until you are returned to **Selection for ...** dialog box. The new variable appears in the **Description** window.

7. Select the variable and click the right arrow ➡ to move the selected variable over to the **Selection** window, and save the query. You are now ready to import the query in Framework Manager.

Creating a Package in Framework Manager

To create a package in Framework Manager you must

- Import the SAP BW metadata using the MetaData wizard

- Create a package

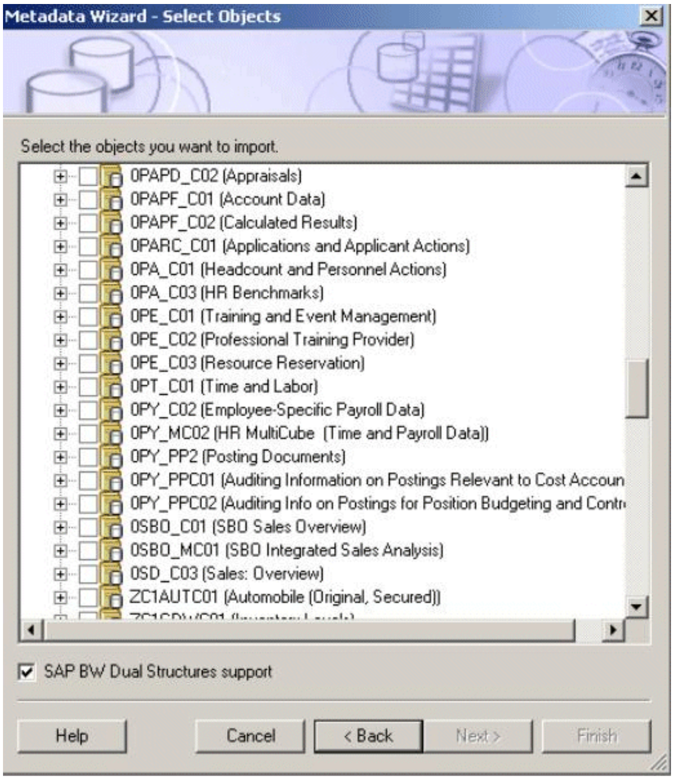
Importing Using the Metadata Wizard

Framework Manager imports the SAP BW query into a model, and defines a package that it exports to Content Manager. When importing, note the following:

- The dimensions selected in the SAP BW query are available in the **Dimension Folders** in the **Import** dialog box.
- Each dimension will contain at least one hierarchy.
- Always select the primary hierarchy whose name matches the hierarchy.
- If other hierarchies are available, select one that gives the desired set of levels within the hierarchy.
- Framework Manager imports time dimensions into the model from the SAP BW data source only if a configuration parameter is turned on. Setting the configuration as a time dimension is a global entry; every imported dimension will then be treated as time strings.

Steps

1. In Framework Manager, click **Create a new project.**
2. Complete the fields in the **New Project** dialog box. Click **OK**.
3. Complete the steps in the **Metadata Wizard**. When prompted to select a data source, if you need to create a new data source, click **New...**
4. In the **Select Objects** page, locate the query that you defined in SAP BW query Designer in the previous stage (p. 362). Scroll the list for the technical name that you provided when you created the variable. The folder structure is as follows: Hierarchies > Level definitions > Query Item definitions.



Metadata Wizard - Select Objects

Select the objects you want to import.

- □ 0PAPD_C02 (Appraisals)
- □ 0PAPF_C01 (Account Data)
- □ 0PAPF_C02 (Calculated Results)
- □ 0PARC_C01 (Applications and Applicant Actions)
- □ 0PA_C01 (Headcount and Personnel Actions)
- □ 0PA_C03 (HR Benchmarks)
- □ 0PE_C01 (Training and Event Management)
- □ 0PE_C02 (Professional Training Provider)
- □ 0PE_C03 (Resource Reservation)
- □ 0PT_C01 (Time and Labor)
- □ 0PY_C02 (Employee-Specific Payroll Data)
- □ 0PY_MC02 (HR MultiCube (Time and Payroll Data))
- □ 0PY_PP2 (Posting Documents)
- □ 0PY_PPC01 (Auditing Information on Postings Relevant to Cost Accoun
- □ 0PY_PPC02 (Auditing Info on Postings for Position Budgeting and Contr
- □ 0SBO_C01 (SBO Sales Overview)
- □ 0SBO_MC01 (SBO Integrated Sales Analysis)
- □ 0SD_C03 (Sales: Overview)
- □ ZC1AUTC01 (Automobile (Original, Secured))

☑ SAP BW Dual Structures support

5. Select the main query items that directly relate to the level. That is, those labeled (**Key**), (**Name**), and so on.

Tip: Secondary or additional attributes are removed on import to Transformer. Only items that are needed are imported. However, to improve performance, we recommend that you do not select secondary or additional attributes. If you select all the attributes here, you can exclude unwanted query items when publishing the package.

6. Complete the remaining screens in the **Metadata Wizard**, accepting the default values, and click **Next**. This will generate dimensions and import the metadata.

7. At the final wizard screen, verify the results, and click **Finish**.

You have imported the SAP BW metadata into Framework Manager.

Creating a Package

When creating the package for publishing to Content Manager, hide the primary hierarchy in those dimensions where you imported two hierarchies. The primary hierarchy is necessary, and must be in the package for querying to work correctly. You can hide the hierarchy if you don't want it visible.

Steps

1. Click the **Packages** folder, and from the **Actions** menu, click **Create**, **Package**.

2. In the **Provide Name** page, type the name for the package and, if you want, a description and screen tip. Click **Next**.

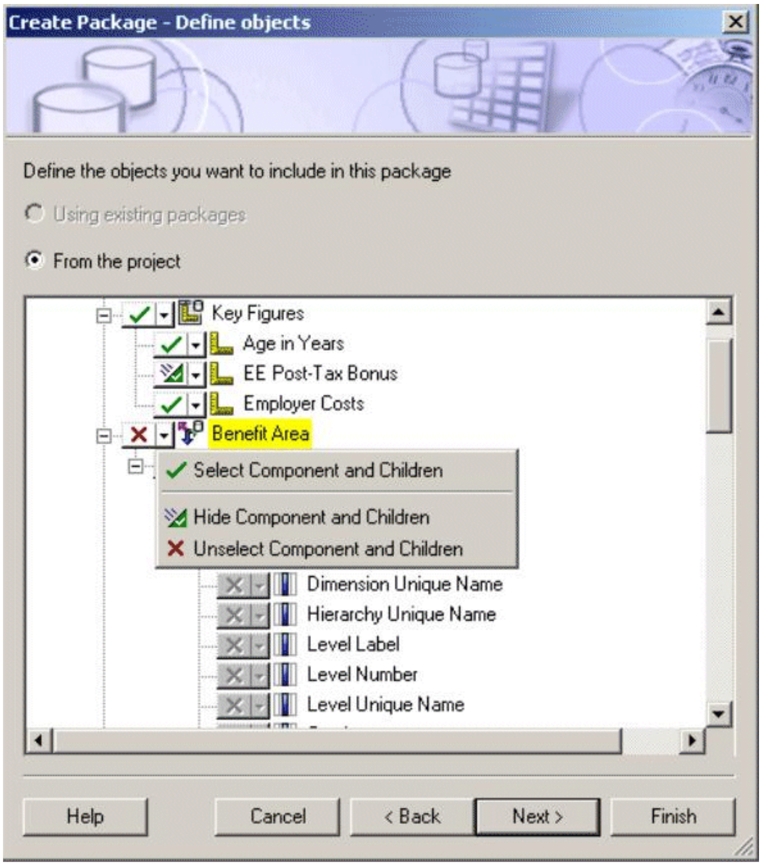
3. Select the query that you imported in the previous section.

   For more information, see "Creating a BW Query in SAP Business Explorer Query Designer" (p. 362).

4. In the **Define objects** page, when hiding or excluding child objects from the package, you must select each of them individually. Excluding parent objects also exclude all of its children. Note that excluding (or unselecting) many objects from larger cubes will require a significant amount of time.

   Note: Framework Manager supports ctrl+shift and alt+shift functionality. Use these keystrokes to select multiple objects that you wish to include or hide in the cube. For example, if you wish to only include two items in a large branch, select the entire branch, then use ctrl+shift to de-select the items you wish to include, and hide the remaining selected items.

   For more information about including, excluding and hiding objects, see "Create or Modify a Package" in the Framework Manager *User Guide*.



Create Package - Define objects

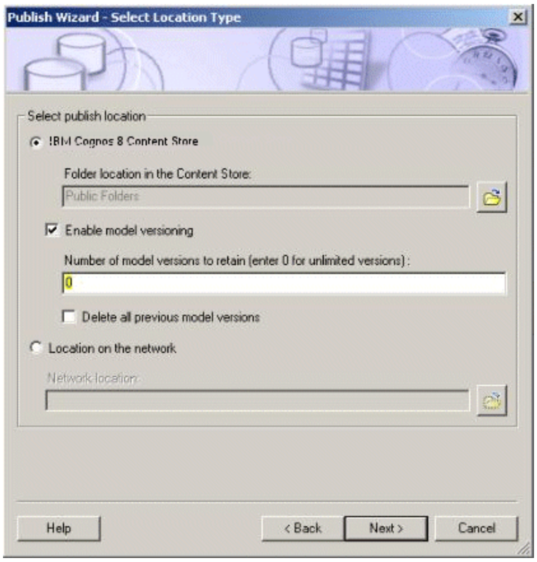
Define the objects you want to include in this package

○ Using existing packages

● From the project

5. Choose whether to use the default access permissions for the package:
   - To accept the default access permissions, click **Finish**.
   - To set the access permissions, click **Next**.

6. When you are prompted to open the **Publish Package Wizard**, click **Yes**.



Publish Wizard - Select Location Type

Select publish location

● IBM Cognos 8 Content Store

Folder location in the Content Store: Public Folders

☑ Enable model versioning

Number of model versions to retain (enter 0 for unlimited versions): 0

☐ Delete all previous model versions

○ Location on the network

Network location:

7. Select the default values, and click **Publish**. This will publish the package to the content store, and will allow you to access the package in Transformer.

8. At the final screen verify the results, and click **Finish**.

You are now ready to create a model in Transformer. For more information on creating a package, see "Create or Modify a Package" in the *Framework Manager User Guide*.

Creating a Model in Transformer

Use Transformer to access a published SAP-based package and use it as a data source to create a model. After the model is created, you can create PowerCubes for use with the desired IBM Cognos 8 component, accessing the dimensional and fact data from the original SAP BW source. In addition, you can combine the SAP metadata in a Transformer model with other corporate metadata or personal sources provided you have the necessary matching key information to join the data during cube building.

When you create the Transformer model, you must use the **Insert Dimension from Package** wizard rather than the **New Model Wizard**. You use the **Insert Dimension from Package** wizard because it

- creates a single query for each dimension and for the facts.
- imports facts and dimensions in the same manner as dimensionally-modeled relational models. That is, facts and dimensions are imported at the same time.
- ensures that the scope is set properly between the dimensions and facts.
- populates the dimension with the appropriate business key and caption information.
- only imports the necessary items from the BW package required for cube building, when the metadata is imported. This reduces the number of attributes and keeps the data volumes to only the necessary items for cube building.

If you want to define business rules, do so in the Transformer model rather than in Framework Manager. Calculations that you define in Framework Manager are not imported into Transformer.

Steps

1. In Transformer, click **Create a new model**.
2. In the **New Model Wizard**, click **Cancel**.
3. With the **Dimension Map** pane selected, from the **Edit** menu, click **Insert Dimension from Package**.
4. Click **Browse** to open the metadata browser.
5. In the **Browse Metadata** dialog box, select the package that contains your SAP BW query and click **OK**.
6. In the **Insert Dimension from Package** dialog box, click **Finish.**
7. In the **Select Dimension and Measures from Package** dialog box, click the dimensions and measures to include in the data source.

   Select a query item that will provide the dates for the PowerCube. Note that the dates for the PowerCube can be derived entirely from the transaction data.

8. If there are errors or warnings, you are notified. In the **Data Sources** pane, expand the package to view the data source queries and query items. Key figures or measures appear in the **Measures** pane.

   Ensure that the aggregation rule for each measure is correctly defined within Transformer to align as closely as possible with the aggregation rule defined in SAP BW.

   It is recommended that the storage type for all measures be set to 64-bit floating point.

   For the root level of each characteristic (dimension), ensure it is marked as unique.

   SAP BW presentation hierarchies may contain ragged paths, typically in association with the "not assigned" and "#" nodes in the hierarchy. The gaps in these hierarchies produce blanks at the associated level in the Transformer hierarchy. In Transformer, it is possible to define the text that should be used for blanks (the default text is "<blank>"). A best practice is to define a more appropriate text for blank entries for all such levels.

9. If you want to add another query, repeat steps 3 to 7.

10. Save the model.

You can now use this model to create PowerCubes for use with the desired IBM Cognos 8 component, accessing the dimensional and fact data from the original SAP BW data source. For more information, see the section "Create a Model" in the Transformer *User Guide*.

Working with SAP BW Data Using Externalized CSV Files in Framework Manager

When you externalize query subjects and dimensions into formats that you can use in other applications, there are special considerations. When extracting data from SAP BW using Framework Manager, you must understand the distinction that Framework Manager makes between different types of dimensions. Each type of dimension exhibits a different behavior when it is externalized, and can be modified before externalizing.

In Transformer versions 7.x, 8.1, and 8.2, you can leverage your SAP BW data using a Framework Manager package in which the query subjects and dimensions are externalized using CSV files. Transformer can use the CSV files as a data source to create a model and generate PowerCubes. CSV files are also supported in Transformer version 8.3 but it is recommended that you use package support for dimensional data and CSV files for fact data.

In Transformer version 8.4, using Framework Manager packages published to Content Manager is the preferred method to leverage SAP BW data. For general information about creating packages, see "Create or Modify a Package" in the Framework Manager *User Guide*. For SAP-specific information about creating packages, see "Work with SAP BW Data Using a Package in Framework Manager" (p. 361).

Extract Size

The **Extract Size** data source property within Framework Manager controls the amount of data retrieved from SAP BW at any one time.

If this setting is negative, zero, or empty, a single query is issued to SAP BW to extract the characteristic data.

If this setting is a positive value, Framework Manager issues multiple queries to SAP BW, each of which returns approximately the number of megabytes specified by the **Extract Size** property.

This feature can reduce the overall size of the query result on the SAP BW server. Overall query execution may take longer, but for large characteristics, not using this feature may result in consumption of a user's allotted memory space on the SAP BW server.

The entire data for a characteristic dimension will be in memory within Framework Manager prior to the production of an extract file. It is important that only the required query items be extracted from SAP BW to ensure that an extract does not fail due to memory allocation errors within Framework Manager.

Model query subjects are extracted using the same mechanism by which queries are executed within IBM Cognos 8. Therefore, the **Extract Size** property has no effect on the query execution.

Measure Dimensions

When extracting a measure dimension, you should create a model query subject containing the measures that you want. You should include the business key query item from each of the levels of each dimension, depending on the level of granularity that you are trying to achieve.

For information about externalizing model query subjects, see "Framework Manager Considerations" (p. 374).

Characteristic Dimensions

Characteristic dimensions are externalized independent of the type of SAP BW data source, such as InfoCube or SAP BW query.

Framework Manager uses a single approach to externalize all dimensions that do not contain fact query items. In these cases, the extract size configuration setting is used to control the manner in which data is extracted from SAP BW.

Note: Model query subjects are externalized in a different manner, regardless of whether they contain fact query items or not. For information about externalizing model query subjects, see "Framework Manager Considerations" (p. 374).

Key Figures Dimensions from an SAP BW InfoCube

When externalizing the key figures dimension from a model based on an InfoCube, Framework Manager uses exactly the same approach as used for externalizing model query subjects.

For an InfoCube containing more than a few thousand transactions, externalizing an InfoCube directly from Framework Manager can easily exceed both time and memory limits on either the client or server. In such cases, it is highly recommended that an SAP BW query be used as the basis for externalizing the SAP BW metadata.

Key Figures Dimensions from an SAP BW Query

Using a BEx query as the basis for externalizing key figures from an SAP BW data source is, in most cases, the best approach. By using a BEx variable to break the data of the key figure dimension into manageable sections, arbitrarily large volumes of transaction data can be extracted from SAP BW.

Note, however, that this approach incurs some restrictions as to what can be extracted from SAP BW, and how it can be extracted. The remainder of this section describes how an SAP BW query is used to extract data from SAP BW, including all known restrictions and limitations.

SAP BW Query Requirements

For the remainder of this section, we assume that an SAP BW query is being used as the basis for externalizing the data, not as the basis for reporting, and not with the intent of exceeding the memory and time limitations associated with extracting data directly from an InfoCube.

It is not possible to externalize an arbitrary SAP BW query. An SAP BW query must adhere to the following restrictions if you want to externalize it:

- Set the characteristic display to **Key**. Setting the display to anything else may result in incorrect data.

  To change what appears for a characteristic, right-click the characteristic and click **Properties**. In the **Properties of Characteristic** dialog box, change the **Display As** value to **Key**.

  We strongly recommend that you use **Key**.

- To reduce data volumes, as well as the amount of aggregation performed by the SAP BW server, we strongly recommend that summarization for all characteristics in the query be disabled in its property sheet.

  To disable summarization for a characteristic, right-click the characteristic along the edge of the SAP BW query and click **Properties**. In the **Properties** dialog box, set the **Suppress Results Rows** value to Always.

- If at least one characteristic in an InfoQuery is displayed as something other than Key, then summarization for all characteristics must be suppressed.

- The query must not contain the Currency/Unit characteristic.

- None of the characteristics may be assigned a display hierarchy, either explicitly or by a variable.

- If a characteristic is included in an SAP BW query as a free characteristic, no values will appear for that characteristic in the key figures extract.

  A filter on a free characteristic acts as a filter on the data returned by SAP BW. It is an efficient mechanism for defining a subset of an InfoCube.

  Such a filter may also be applied to a characteristic along an axis of an SAP BW query, in which case the filtered values appear in the key figures extract.

- All key figures in the SAP BW query must be numeric.

- The values of each key figure should be in a single currency. A variable should not be used to drive the assignment of a target currency.

- Include in the SAP BW query only those characteristics which are to be extracted using Framework Manager. Including unnecessary characteristics increases the volume of data transferred from SAP BW, thus affecting performance.

Guidelines and Constraints When Working with SAP BW Cubes

You must use CSV files when importing metadata from SAP BW cubes. For performance reasons, we recommend that you filter on geography, time periods, or some other dimension that limits the amount of data retrieved. Remember to apply your dimension filter to the related dimensions and their fact tables (measures). For more information, see "Create a Filter" in the Framework Manager *User Guide*.

Because SAP BW cubes are multidimensional, rollups are applied at the source. If you change the rollup type after importing the data into Transformer, your results will not be valid.

Missing data or metadata that is out-of-scope for a particular measure may yield different results, depending on the context. You may see:

- NULL values
- # symbols
- REST_H
- Not assigned

Because such duplicate tokens can cause problems in Transformer, in unique levels for example, we recommend that you assign filters to the dimension so that they do not appear in the imported data.

Finally, remember to select only those query items needed to generate your filtered data.

Framework Manager Considerations

When extracting the measure dimension from an SAP BW query, the **Extract Size** property of the data source controls the amount of data retrieved from the SAP BW server at one time. Model query subjects are externalized in a different manner, regardless of whether they contain fact query items or not. In this scenario, the setting has no affect on the SAP BW server, but it does limit the amount of memory Framework Manager allocates at any one time to retrieve the data.

Note that filters defined on the key figures dimension are not enforced when extracting data from an SAP BW query. To obtain performance benefits of extracting data from an SAP BW query, filters must be defined in an SAP BW query.

In addition, any calculations defined within the key figures dimension are ignored. These may be defined either within the SAP BW query in BEx, or in a model query subject in Framework Manager.

Each characteristic extracted must contain at least one query item from the lowest level of its hierarchy (if there is one) to provide linkage with the key figures extract. You should include the business key query item from each of the levels of each dimension, depending on the level of granularity that you are trying to achieve.

Use of Variables to Externalize Key Figures from an SAP BW Query

The volume of transactions within an SAP BW query is such that, in most cases, the use of a single query to extract the data from SAP BW will exceed the memory allocated to a user on an SAP BW server. In Framework Manager, you can use a single optional variable to extract fact data from an SAP BW query in reasonably sized sections.

To use this feature, one characteristic included in the SAP BW query (but not included as a free characteristic) is assigned a variable that conforms to the following restrictions:

- It must be a single value.
- It must be optional.
- It must not have a default value.
- It can be defined on the characteristic or a presentation hierarchy.

If an SAP BW query contains such a variable and the key figures dimension is externalized, Framework Manager runs a query for each possible value associated with a variable. Thus, by choosing an appropriate characteristic, the key figures dimension can be extracted without exceeding the memory restrictions of either the client or server. Memory caches on the client and server are flushed after each query.

If a presentation hierarchy is used to drive the creation of extract sections, it is important that the values for a variable be obtained from a single level in the hierarchy, otherwise the extract will contain data summarized at different levels. To restrict the values for a variable to a single level of a hierarchy, edit the Level Restriction of the variable in Framework Manager. For example, using a value such as "2:2" indicates that only values from the second level of the hierarchy are to be used (level 0 is the root of a hierarchy).

In the presence of an SAP BW query with one such variable, the value of the variable is reset after each query.

If an SAP BW query contains anything more than a single variable, or one that is defined differently than described above, Framework Manager does not attempt to use a variable to break the extraction of the key figures dimension into smaller sections.

Workaround for Problems Encountered While Externalizing

When externalizing a data source from Framework Manager, you may encounter an authentication error if

- the model is published to Content Manager
- externalizing the data takes longer to perform than the timeout period assigned to passports within IBM Cognos Configuration

Users are not prompted to re-enter their authentication credentials.

If an error occurs, the externalized data is still complete and valid. However, if the modeler chooses to actually publish the model, the modeler must re-authenticate and re-publish the model, but without externalizing the data.

Another solution is to publish the model to the network, in which case the authentication error does not occur.

Building PowerCubes from SAP BW Data

You can build IBM Cognos PowerCubes from SAP BW data. There are guidelines to consider for both Framework Manager and Transformer.

Framework Manager Guidelines

When externalizing data for the purpose of creating one or more PowerCubes, keep these considerations in mind.

- The extract of each characteristic must have a common key query item that is equivalent to a surrogate key query item in the key figures extract.
- For an extract based on an SAP BW query, it is strongly recommended that all characteristics be displayed as Key in the SAP BW query.
- If a characteristic does not have a presentation hierarchy, or a new one is desired, extract one or more query items that can form the basis for levels in a hierarchy.
- During the import of SAP BW metadata into a model that will extract data, limit the model to only those query items that are absolutely required to build a PowerCube. This will improve data extract performance.
- A practical limit for PowerCubes is 2,000,000 categories (values) for a dimension (characteristic).

Transformer Guidelines

When using the SAP BW data that you extracted from Framework Manager, keep these considerations in mind.

- In Transformer version 8.3, you can insert regular dimensions from SAP data sources directly from a IBM Cognos 8 data source, using the **Insert dimension from package** option.
- Using the model wizard in Transformer, insert a data source of type Delimited-Field Text With Column Titles and start by selecting the CSV file. Do not run auto-design.
- Drag all the key figure columns from the Data Sources pane into the Measures pane. Ensure that the aggregation rule for each measure is correctly defined within Transformer to align as closely as possible with the aggregation rule defined in SAP BW.
- It is recommended that the storage type for all measures be set to 64-bit floating point.
- Using the date wizard, select a query item that will provide the dates for the PowerCube. Note that the dates for the PowerCube can be derived entirely from the transaction data.
- Insert the various CSV files corresponding to the characteristics that were externalized using Framework Manager.

  Each CSV file contains a column that corresponds to a column in the key figures CSV file. By right-clicking the various columns and editing the column properties, ensure the columns that provide the linkage between a characteristic and the key figures have the same name. For example, if a key figure column is named Customer and the corresponding column in the customer CSV file is named Customer - Key, then the name of the column in the key figures CSV file can be changed to Customer - Key.
- For each characteristic, create a new dimension, using the key columns, or other attributes of a characteristic, to drive the levels of the dimension. For each level, ensure that the properties for the label, short name, and description are assigned source columns, if applicable.

- For the root level of each characteristic (dimension), ensure it is marked as unique.
- SAP BW presentation hierarchies may contain ragged paths, typically in association with the "not assigned" and "#" nodes in the hierarchy. The gaps in these hierarchies produce blanks at the associated level in the Transformer hierarchy.

In Transformer, it is possible to define the text that should be used for blanks (the default text is "<blank>"). A best practice is to define a more appropriate text for blank entries for all such levels.

Appendix F: IBM Cognos 8 Transformer Expression Editor

The Transformer version 8.x expression editor is not the same expression editor as the one used in Framework Manager or the Web studios.

The expression editor can be invoked from a number of places within Transformer. The functionality varies depending upon whether you are creating a calculated dimension, calculated category, calculated column or calculated measure.

An expression is any combination of operators, constants, functions, and other components that evaluates to a single value. You build expressions to create calculations. A calculation is an expression that you use to create a new value from existing values contained within a data item.

An expression can include

- functions, such as Day and Months-Between

  The **Functions** folder in the **Available Components** box contains functions provided by Transformer. These are pre-defined calculations, such as string, numeric, and date calculations, that are designed to operate on various data types.

- constants, such as numbers, strings, and date

  Constants are typed directly into the expression once the type of value is determined. Remember to press Enter to complete the entry.

  The expression editor is context-sensitive, so it presents only options that are relevant at each stage of expression building.

- operators, such as +

  Operators specify what happens to the values on either side of the operator. There are four types:

  - mathematical operators perform mathematical operations on two parts of an expression (for example: +, -, *, /, ^)
  - string operators concatenate two character strings (+)
  - logical operators define a relationship between two parts of an expression (for example: and, not, or)
  - comparison operators filter data by comparing one or more values that you enter against the values in the expression (for example: like, =)

  Note: Transformer supports mathematical operators and string concatenate operators. It also supports logical operators and comparison operators, but only in an if-then-else conditional construct.

- the if-then-else conditional construct

You can build expressions in four functional areas in Transformer:

- calculated columns in a data source
- calculated categories
- calculated measures
- calculations that apply to a set of categories or levels within a dimension

The components that are available to build an expression depend on:

- the type of calculation
- the type of data class (numeric, date, or text)
- the type of data column
- whether the calculation includes an if-then-else conditional construct

As you add or type each component in the expression, the syntax is evaluated against mathematical rules. The Tips window of the expression editor provides information about the selected component.

Building Expressions

To build an expression quickly, you can type it in the expression editor without selecting components from the left pane, or you can type the parts of the expression that you know and add components.

The expression editor validates the expression as you build it. If the expression is not mathematically valid, the expression editor highlights the errors in the expression, and error messages appear in the tip box.

Tip: Before typing a function into the expression editor, confirm that it is listed in the left pane. This will ensure that your application supports that function.

For information on functions, see "Functions" (p. 382). For information on applying or working with calculations, see the appropriate product documentation.

Steps

1. From the left pane, locate and select the component you want.
2. Click the arrow button to add the component to the expression.
3. Continue adding components until your expression is complete.
4. Click **OK**.

Tips

- To add components quickly to the expression, double-click the component in the left pane.
- To move around in the **Available components** and **Expression boxes** you can use the mouse, or:
    - press the space bar to select a component

  - press the up and down arrows to move within the **Available Components** box
  - press the left and right arrows to move within the **Expression definition** box

- Selecting **String** inserts two quotation marks and positions the cursor between them. You enter the string between the quotation marks.
- Selecting **Number** inserts the number 0. You overwrite it with the number you want included in the expression.
- Selecting **Date** inserts the current date in quotation marks. You can overwrite it with other data values.

The components of an expression can include functions, summaries, values, and operators.

Building an If-Then-Else Calculated Expression

Your models can include calculated measures that use if-then-else expressions. For example, you can define the following conditional expression:

```
if ("Net Income"<100000) then ("Gross Profit"*1.25) else
NULL
```

You can also define an if-then-else calculation to avoid division by zero:

```
if ("MEASURE_VALUE_A"=0) then ("MEASURE_VALUE_B"=NULL) else ("MEASURE_VALUE_B"/
"MEASURE_VALUE_A")
```

Steps

1. In the Expression Editor, begin your calculated measure definition by clicking the `if` operator.

   An opening parenthesis is inserted next.

2. Enter the rest of the expression as follows:

   - Type a conditional expression that resolves to `True` or `False`.
   - Double-click a closing parenthesis.
   - After the automatically inserted expression `"then ("`, type the result to show if the boolean expression is `True`.
   - Double-click a closing parenthesis.
   - After the automatically inserted expression `"else ("`, type the result to show if the boolean expression is `False`.
   - Double-click a closing parenthesis and click **OK**.

   If your expression is valid, the definition is saved for that measure.

3. If an error appears, try again, using a mathematically correct form.

   Tip: Unlike calculated columns, calculated measures support `isnull( )` expressions that resolve to `True`, which is the condition that arises with a missing value. You can use this capability to avoid divide-by-zero overflow errors.

4. Build the cube, open it in your reporting application, and confirm that the results correctly reflect the if-then-else condition you were trying to model.

Functions

A function is a subroutine that returns a single value. You can use functions to create calculations and conditions to filter data. Functions are similar to operators in that they manipulate data items and return a result. Functions differ from operators in the format in which they appear with their arguments. This format allows them to operate with zero, one, two, or more arguments:

```
function (argument, argument, ...)
```

Functions are of these general types:

Function	Description
Date Functions	Accepts numeric input and returns a value that is a date.
Numeric Functions	Accepts numeric input and returns numeric values.
Aggregate Functions	Executes a predefined function and returns an aggregate value.
Text Functions	Accepts character input and can return both character and number values.

Date Functions

Date functions return a value that is either a date or a number that relates to a date.

Function	Description
Add-Days	Returns a datetime value resulting from adding a number of days to a date.
Add-Months	Returns a datetime value resulting from adding a number of months to a date.
Add-Years	Returns a datetime value resulting from adding a number of years to a date.
Age	Returns age as a day-month-year interval by subtracting a specified date from today's date.
Date-to-Days-from-1900	Returns the number of days since Jan 1, 1900 inclusive. The value returned is negative if the date is before 1900.

Function	Description
Day	Returns a numeric value for the day of the month from 1 to 31, from a date, datetime value, or interval.
Days-to-End-of-Month	Returns the number of days to the last day of the month from a date or datetime value.
First-of-Month	Returns the first day of the month from a date or datetime value.
Last-of-Month	Returns the last day of the month from a date or datetime value.
Month	Returns the month number as an integer from 1 to 12, from a date or datetime value.
Months-Between	Returns the number of months between two dates. If the first date is later than the second date, then the result is a negative number.
Today	Returns the current date according to the date set on your computer.
Year	Returns the year from the date.
Years-Between	Returns the number of years from one date to another date.

Add-Days

Returns a datetime value resulting from adding a number of days to a date.

Syntax

```
add-days (date_exp | datetime_exp, integer_exp)
```

Examples

```
add-days (today(), 10)
```

Returns the result: 03/30/2007 00:00

```
add-days (today(), -10)
```

Returns the result: 03/10/2007 00:00

Add-Months

Returns a datetime value resulting from adding a number of months to a date.

Syntax

```
add-months (date_exp | datetime_exp, integer_exp)
```

Examples

```
add-months (today(), 10)
```

Returns the result: 01/20/2007 00:00

```
add-months (today(), -10)
```

Returns the result: 05/20/2007 00:00

Add-Years

Returns a datetime value resulting from adding a number of years to a date.

Syntax

```
add-years (date_exp | datetime_exp, integer_exp)
```

Examples

```
add-years (today(), 10)
```

Returns the result: 03/20/2017 00:00

```
add-years (today(), -10)
```

Returns the result: 03/20/1997 00:00

Age

Returns age as a month-day-year interval by subtracting a specified date from today's date.

Syntax

```
age (date_exp | datetime_exp)
```

Example

```
age (1996-08-19)
```

Returns the result: 02/01/0002 00:00

Note

Date-to-Days-from-1900

Returns the number of days since Jan. 1, 1900 inclusive.The value returned is negative if the date is before 1900.

Syntax

```
date-to-days-from-1900 (date_exp | datetime_exp)
```

Example

```
date-to-days-from-1900 (1998-03-20)
```

Returns the result: 35873

Day

Returns a numeric value for the day of the month from 1 to 31, from a date, datetime value, or interval.

Syntax

```
day (date_exp | datetime_exp | interval_exp)
```

Examples

```
day (2007-03-20)
```

Returns the result: 20

```
day (2007-03-20 18:22:00.000)
```

Returns the result: 20

```
day (20 00:00:00.000)
```

Returns the result: 20

Days-to-End-of-Month

Returns the number of days to the last day of the month from a date or datetime value.

Syntax

```
days-to-end-of-month (date_exp | datetime_exp)
```

Example

```
days-to-end-of-month (2007-03-20)
```

Returns the result: 11

First-of-Month

Returns the first day of the month from a date or datetime value. The datetime value is converted from a date_exp value to a date with the same year and month, but the day is set to one.

Syntax

```
first-of-month (date_exp | datetime_exp)
```

Example

```
first-of-month (2007-03-20)
```

Returns the result: 03/01/2007 00:00

Last-of-Month

Returns the last day of the month from a date or datetime value.

Syntax

```
last-of-month (date_exp | datetime_exp)
```

Example

```
last-of-month (2007-03-21)
```

Returns the result: 03/31/2007 00:00

Month

Returns the month number as an integer from 1 to 12, from a date or datetime value.

Syntax

```
month (date_exp | datetime_exp)
```

Examples

```
month (2007-03-21)
```

Returns the result: 3

```
month (2007-03-21 09:21:00.000)
```

Returns the result: 3

Months-Between

Returns the number of months between two dates. If the first date is later than the second date, then the result is a negative number. This function does not round months; the days and time portions of the difference are ignored.

This function is processed only on an IQD data source.

Syntax

```
months-between (date_exp_1 | datetime_exp_1, date_exp2 | datetime_exp2)
```

Examples

```
months-between (2007-03-21, add-months (2007-03-21, 4))
```

Returns the result: 4

```
months-between (2007-01-31, 2007-02-01)
```

Returns the result: 0

```
months-between (2007-01-31, 2007-03-21)
```

Returns the result: 1

Today

Returns the current date according to the date set on your computer.

Syntax

```
today ()
```

Example

```
today ()
```

Returns today's date as the result.

Year

Returns the year from the date.

Syntax

```
year (date_exp | datetime_exp)
```

Example

```
year (2007-12-10)
```

Returns the result: 2007

Years-Between

Returns the difference between the year values of date_exp1 and date_exp2. If date_exp1 is later than date_exp2, the result will be a negative number.

Syntax

```
years-between (date_exp1 | datetime_exp1, date_exp2 | datetime_exp2)
```

Example

```
years-between (2005-03-21, 2007-03-21)
```

Returns the result: 2

Numeric Functions

Accepts numeric input and returns numeric values.

Function	Description
Ceiling	Returns a number rounded to the next highest integer.
Floor	Returns a number rounded to the next lowest integer.
Integer-Divide	Returns the integer obtained from truncating the result of an integer divided by a second integer.
Mod	Returns the remainder (modulus) of an integer divided by a second integer.
Number-to-String	Returns a string from a number.
Round-Down	Returns a number rounded down.
Round-Near	Returns a number rounded to the nearest value.
Round-Up	Returns a number rounded up.
Round-Zero	Returns a number rounded toward zero.
Sqrt	Returns the square root of a positive number.

Ceiling

Returns a number rounded to the next highest integer.

Syntax

```
ceiling (numeric_exp)
```

Examples

```
ceiling (-1.23)
```

Returns the result: -1

```
ceiling (1.23)
```

Returns the result: 2

Floor

Returns a number rounded to the next lowest integer.

Syntax

```
floor (numeric_exp)
```

Examples

```
floor (-1.23)
```

Returns the result: -2

```
floor (3.45)
```

Returns the result: 3

Integer-Divide

Returns the integer obtained from truncating the result of an integer divided by a second integer.

Syntax

```
integer-divide (integer_exp1, integer_exp2)
```

Examples

```
integer-divide (10, 20)
```

Returns the result: 0

```
integer-divide (20, 6)
```

Returns the result: 3

Mod

Returns the remainder (modulus) of an integer divided by a second integer. If the second integer is zero, Transformer issues a divide by zero error.

Syntax

```
mod (integer_exp1, integer_exp2)
```

Example

```
mod (245,3)
```

Returns the result: 2

Number-to-String

Returns a string from a number. If the number is negative, a minus sign (-) precedes the string. If the number is a real number, only the truncated integer part of the number is converted to a string.

Syntax

```
number-to-string (numeric_exp)
```

Examples

```
number-to-string (12345)
```

Returns the result: 12345

```
number-to-string (12345.678)
```

Returns the result: 12345

Round-Down

Returns a number rounded down.

The integer_exp value determines the position that is rounded. A positive integer_exp acts on the digits to the right of the decimal point. A negative integer_exp acts on the digits to the left of the decimal point. An integer_exp value of zero rounds the number and removes the decimal places.

Syntax

```
round-down (numeric_exp, integer_exp)
```

Examples

```
round-down (-113.6667, 0)
```

Returns the result: -114

```
round-down (-113.6667, 1)
```

Returns the result: -113.7

```
round-down (-113.6667, -1)
```

Returns the result: -120

```
round-down (-113.6667, -2)
```

Returns the result: -200

```
round-down (366.2162, 0)
```

Returns the result: 366

```
round-down (366.2162, 1)
```

Returns the result: 366.2

```
round-down (366.2162, -1)
```

Returns the result: 360

```
round-down (366.2162, -2)
```

Returns the result: 300

Round-Near

Returns a number rounded to the nearest value.

The integer_exp value determines the position that is rounded. A positive integer_exp value acts on the digits to the right of the decimal point. A negative integer_exp value acts on the digits to the left of the decimal point. An integer_exp value of zero rounds the number and removes the decimal places.

Syntax

```
round-near (numeric_exp, integer_exp)
```

Examples

```
round-near (-113.6667, 0)
```

Returns the result: -114

```
round-near (-113.6667, 1)
```

Returns the result: -113.7

```
round-near (-113.6667, -1)
```

Returns the result: -110

```
round-near (-113.6667, -2)
```

Returns the result: -100

```
round-near (366.2162, 0)
```

Returns the result: 366

```
round-near (366.2162, 1)
```

Returns the result: 366.2

```
round-near (366.2162, -1)
```

Returns the result: 370

```
round-near (366.2162, -2)
```

Returns the result: 400

Round-Up

Returns a number rounded up.

The An integer_exp value determines the position that is rounded. A positive An integer_exp value acts on the digits to the right of the decimal point. A negative An integer_exp value acts on the digits to the left of the decimal point. An integer_exp value of zero rounds the number and removes the decimal places.

Syntax

```
round-up (numeric_exp, integer_exp)
```

Examples

```
round-up (-113.6667, 0)
```

Returns the result: -113

```
round-up (-113.6667, 1)
```

Returns the result: -113.6

```
round-up (-113.6667, -1)
```

Returns the result: -110

```
round-up (-113.6667, -2)
```

Returns the result: -100

```
round-up (366.2162, 0)
```

Returns the result: 367

```
round-up (366.2162, 1)
```

Returns the result: 366.3

```
round-up (366.2162, -1)
```

Returns the result: 370

```
round-up (366.2162, -2)
```

Returns the result: 400

Round-Zero

Returns a number rounded toward zero.

The integer_exp value determines the position that is rounded. A positive integer_exp value acts on the digits to the right of the decimal point. A negative integer_exp value acts on the digits to the left of the decimal point. An integer_exp value of zero rounds the number and removes the decimal places.

Syntax

```
round-zero (numeric_exp, integer_exp)
```

Examples

```
round-zero (-113.6667, 0)
```

Returns the result: -113

```
round-zero (-113.6667, 1)
```

Returns the result: -113.6

```
round-zero (-113.6667, -1)
```

Returns the result: -110

```
round-zero (-113.6667, -2)
```

Returns the result: -100

```
round-zero (366.2162, 0)
```

Returns the result: 366

```
round-zero (366.2162, 1)
```

Returns the result: 366.2

```
round-zero (366.2162, -1)
```

Returns the result: 360

```
round-zero (366.2162, -2)
```

Returns the result: 300

Sqrt

Returns the square root of a positive number.

Syntax

```
sqrt (numeric_exp)
```

Examples

```
sqrt (2)
```

Returns the result: 1.4142135623731

```
sqrt (64)
```

Returns the result: 8

Text Functions

Text functions accept character input and can return both character and numeric values.

Function	Description
Char_Length	Returns the number of characters in a string.
First-Word	Returns the first word in a string.
Left	Returns a specific number of characters, starting at the left of the string.
Lower	Converts uppercase characters to lowercase.
Position	Returns the starting position of a string in a second string.
Reverse	Reverses the characters in a string.
Right	Returns a specific number of characters, starting at the right of the string.
String-to-Integer	Converts a string to an integer.
Substring	Returns a substring from a string.
Trim-Leading	Returns a string with leading spaces removed.
Trim-Trailing	Returns a string with trailing spaces removed.
Upper	Converts lowercase characters to uppercase.

Note: The Pack, Spread and Substitute text functions are not supported in Transformer version 8.x.

Char_Length

Returns the number of characters in a string.

Syntax

```
char_length (string_exp)
```

Examples

```
char_length ('ABCDEFG')
```

Returns the result: 7

```
char_length ('')
```

Returns the result: 0

```
char_length (' ')
```

Returns the result: 1

First-Word

Returns the first word in a string.

Syntax

```
first-word (string_exp)
```

Example

```
first-word ('Cat sat on the mat')
```

Returns the result: Cat

Left

Returns a specific number of characters, starting at the left of the string.

Syntax

```
left (string_exp, integer_exp)
```

Example

```
left ('ABCDEFG', 2)
```

Returns the result: AB

Lower

Converts uppercase characters to lowercase.

Syntax

```
lower (string_exp)
```

Example

```
lower ('ABCDEFG')
```

Returns the result: abcdefg

Position

Returns the starting position of string_exp1 in string_exp2. The first character in a string is at position one.

Syntax

```
position (string_exp1, string_exp2)
```

Examples

```
position ('DEF', 'ABCDEF')
```

Returns the result: 4

```
position ('Z', 'ABCDEFGH'
```

Returns the result: 0

Reverse

Reverses the characters in a string.

Syntax

```
reverse (string_exp)
```

Example

```
reverse ('ABCDEF')
```

Returns the result: FEDCBA

Right

Returns a specific number of characters, starting at the right of the string.

Syntax

```
right (string_exp, integer_exp)
```

Example

```
right ('ABCDEFG', 3)
```

Returns the result: EFG

String-to-Integer

Returns the integer representation of string_exp.

Syntax

```
string-to-integer (string_exp)
```

Example

```
string-to-integer (' 101 tents')
```

Returns the result: 101

Substring

Returns a substring from a string. The first character in the string is at position one.

Syntax

```
substring (string_exp, integer_exp1, integer_exp2)
```

where:

- string_exp is the string from which you want to extract a substring
- integer_exp1 is the position of the first character in the substring

- integer_exp2 is the desired length of the substring

Example

```
substring ('abdefg', 3, 2)
```

Returns the result: de

Trim-Leading

Returns a string with leading spaces removed. For example, if you merge two data items with leading spaces, use trim-leading to eliminate the spaces between them.

Syntax

```
trim-leading (string_exp)
```

Example

```
trim-leading ('  ABC')
```

Returns the result: ABC

Trim-Trailing

Returns a string with trailing spaces removed. For example, if you merge two data items in an expression and the data items have trailing spaces, the spaces between the data items can be eliminated using the trim-trailing function.

Syntax

```
trim-trailing (string_exp)
```

Example

```
trim-trailing ('XYZ  ')
```

Returns the result: XYZ

Upper

Converts lowercase characters to uppercase.

Syntax

```
upper (string_exp)
```

Examples

```
upper ('Auriga')
```

Returns the result: AURIGA

```
upper ('DeEr')
```

Returns the result: DEER

Aggregate Functions

Executes a predefined function and returns an aggregate for a set of values based on the function values.

Note: You can access aggregate functions through calculated categories, calculated measures and dimension calculations. You cannot access them through calculated columns.

Aggregate	Description
Absolute	Converts numbers to their unsigned value.
Average	Returns the average value.
Change	Returns the difference over a linear series of categories by calculating the change between each pair of categories.
Max	Returns the maximum value of selected data items.
Min	Returns the minimum value of selected data items.
Percent	Returns the percent of the total value for selected data items.
Percent-growth	Returns the growth over a linear series of categories by calculating the percentage change between each pair of categories.
Share	Returns the values of categories as a percent share of other target categories.

Absolute

Converts numbers to their unsigned value.

Use when you need positive numbers, or when you need to find the absolute difference between values in a list of positive and negative values.

Syntax

```
absolute (numeric_exp)
```

Examples

```
absolute (-5.3)
```

Returns the result: 5.3

```
absolute (2)
```

Returns the result: 2

Average

Returns the average value of selected data items.

Transformer supports the Average function with numeric data; the only parameters accepted are the numeric expressions and the auto component.

Syntax

```
average (<numeric_exp)
```

Example

```
average (Sales)
```

Returns the result: The average of all Sales values.

Change

Returns the difference over a linear series of categories by calculating the change between each pair of categories. The order of pairs is based on the object ID of each category.

Note: This function is used for automatically generated relative time categories, such as year-to-date (YTD Change). It appears in the list of **Available Components** when you use the expression editor to create a calculated category.

Tip: To use **ToolTip** on the pop-up diagram to show the labels and object IDs of categories, select the **Object Identifier** check box in the **Preferences** properly sheet, (**Titles** tab).

Syntax

```
change (cat_code, cat_code|set|level)
```

Example

A dimension contains a level with Quarters Q1 to Q4. To create change categories for Q1-Q2, Q2-Q3, and Q3-Q4, use the function

```
change ("Quarter")
```

Max

Returns the maximum value of selected data items.

Syntax

```
max (numeric_exp)
```

Example

```
max (Sales)
```

Returns the maximum value of all Sales values.

Min

Returns the minimum value of selected data items.

Syntax

```
min (numeric_exp)
```

Example

```
min (Sales)
```

Returns the minimum value of all Sales values.

Percent

Returns the percent of the total value for the selected data items.

Users can format the percentage data so that it reads as a ratio (for example: .25 vs. 25%).

Syntax

```
percent (numeric_exp)
```

Example

```
percent (sales 07)
```

Returns the percentage of the total sales for 2007 that is attributed to each sales representative.

Sales Rep	Sales 07	Percentage
Bill Gibbons	60646	7.11%
Bjorn Flertjan	62523	7.35%
Chris Cornel	22396	2.63%
Conrad Bergsteige	13500	1.59%

Percent-growth

Returns the growth over a linear series of categories by calculating the percentage change between each pair of categories. The order of pairs is based on the object ID of each category.

Note: This function appears in the list of **Available Components** when you use the expression editor to create a calculated category.

Tip: To use **ToolTip** on the pop-up diagram to show the labels and object IDs of categories, select the **Object Identifier** check box in the **Preferences** properly sheet (**Titles** tab).

Syntax

```
percent-growth (cat_code, cat_code|set|level)
```

Example

A dimension contains a level with Quarters Q1 to Q4. To create growth categories for Q1-Q2, Q2-Q3, and Q3-Q4, use the function

```
percent-growth (Quarter)
```

Share

Returns the values of categories as a percent share of other target categories.

Syntax

```
share (cat_code,cat_code|set|cat_code, set|set|level)
```

Example

A dimension contains the category Color with several descendant categories, such as Black and Chrome, which you define as a set. To create two share categories, representing each of Black and Chrome, as a percentage of share of the parent, use the function

```
share (set, Color)
```

Constants

A constant is a fixed value that you can use in an expression. You can create a data item based on an expression that contains a constant.

Constant	Description
Date	Inserts the current system date and positions the cursor on the first number of the date.
Single Category	Inserts an expression into a category.
Number	Inserts the number zero, which you can replace with a new numeric value.
String	Inserts two quotation marks and positions the cursor between them.

Operators

Operators specify what happens to the values on either side of the operator.

Mathematical, Logical, and String Operators

Transformer supports mathematical operators and string concatenate operators. It also supports logical operators and comparison operators, but only in an if-then-else conditional construct.

Operators	Description
Mathematical Operators (+, -, *, /, ^)	Perform mathematical operations.
And Logical Operator	Returns `True` if the conditions on both sides are true.
Or Logical Operator	Returns `True` if either of the two conditions on both sides is true.
Not Logical Operator	Returns `True` if the condition is false.

Operators	Description
String Operator (+)	Concatenate two strings together.

Mathematical Operators (+, -, *, /, ^)

Performs mathematical operations. The precedence for processing mathematical operators is as follows:

- exponents (^)
- multiplication (*) and division (/)
- addition (+) and subtraction (-)

Mathematical operators that are at the same level of precedence are evaluated from left to right. You can use parentheses to override this order of precedence.

And Logical Operator

Returns `True` if the conditions on both sides are true.

Example

```
Salary < 60000 and Dept = 'Sales'
```

Retrieves the data where salary is less than $60,000 in the Sales department. Precedence is given to logical operators.

Or Logical Operator

Returns `True` if either of the two conditions on both sides is true.

Example

```
Name = 'Smith' or Name = 'Wong'
```

Retrieves the data for the names Smith and Wong.

Not Logical Operator

Returns `True` if the condition is false.

Example

```
not ( "STATE" <> 'CA' )
```

Returns `True` for all states that are not California.

String Operator (+)

Concatenates two strings together.

Syntax

```
Char1 + Char2
```

Example

```
FirstName + LastName
```

Combines FirstName and LastName without spaces between.

Comparison Operators

Comparison operators compare two values and produce the logical value `True` or `False`.

Operators	Description
Comparison Operator Symbols	Compares two values.
Isnull	Determines if a value is undefined in the data.

Comparison Operator Symbols

Compare two values.

Syntax

```
Value1 [=,<,>,<=,>=] Value2
```

Example

```
Qty < 100
```

If the quantity is less than 100, evaluates to `True` and retrieves only those rows.

```
Price > 1000 or Qty <= 3
```

If the price is greater than 1000 or the quantity is less than or equal to 3, evaluates to `True` and retrieves only those rows.

Isnull

Determines if a value is undefined in the data.

Syntax

```
abc isnull
```

Example

```
Telephone Number isnull
```

If there are no telephone numbers, evaluates to `True`. Retrieves only the rows missing telephone numbers.

Glossary

.iqd file

An impromptu query definition file. A file created by Impromptu that contains the definition for a database query.

.mdc file

A multidimensional cube file. An IBM Cognos file that represents a cube. The .mdc file may contain data or may be a pointer file to an OLAP source.

.mdl file

An Exported Model Definition language file. A file that represents a model and is stored in plain text format using Transformer's Model Definition Language (MDL). An .mdl file can be opened in Transformer or any text editor.

.mdp file

A multidimensional partition file. A file that contains one or more partitions from a cube. The .mdp file manages large .mdc files when the cubes exceed a preset number of data records. Partitions are spread evenly across .mdp files, and an additional .mdc file is added to hold the PowerCube metadata.

.py? file

A model file stored in the default binary format. The ? in the extension .py? is replaced by the character used in your version of Transformer.

.qy? file

In Transformer, a temporary file that contains checkpoint entries of each major stage in the cube creation process. Because this file is deleted when the process ends normally, the existence of a .qy? file indicates that Transformer terminated unexpectedly. The ? in the extension .qy? is replaced by the character used in your version of Transformer.

access permissions

Rules defining the access rights to resources. Access permissions can be granted to any combination of namespaces, groups, or users. Examples of resources are reports and folders.

allocation

The distribution of data, specified at a summary level of a dimension, to lower levels. For example, the measures used to forecast quarterly sales revenue can be distributed to the month and day levels.

alternate drill-down path

In a cube, an alternate path within a dimension that leads to child categories. In Transformer, an alternate path traced from the root category, through a drill category, leading to a low-level category.

Certain tasks, such as partitioning and allocating, cannot be performed on alternate drill-down paths.

ancestor category

In a cube, a category above another category along a drill-down path. For example, 2008 is an ancestor category of 2008/Apr. In Transformer, the Categories diagram shows direct ancestors (parents) of a category to the left of the category, linked by a line.

anonymous access

A type of access that allows users and servers to access a server without first authenticating with it.

apex

In Transformer, a customization option that restricts access to selected information in the cube. The apex option omits the ancestors and siblings of a category from the dimension view. The base category in the resulting dimension view is the apexed category. When a cube is created from this dimension or custom view, users see only the apexed category, its descendants, and any special categories that reference them. An apex action can be performed only in the Categories diagram.

association role

A description of the association between an item in a model and a data source. For example, an association may include a source role of Product Number and a label role of Product Name.

attribute

In dimensional models, a property that provides qualitative information about members of a level in a dimension. For example, the Store level within the Retailer dimension might have properties such as address or retail space. In general, dimensional attributes do not have measure values or rollups associated with them, but are used to locate or filter members.

In relational models, a query item that is not a measure or identifier. When a query item is an attribute, it is not intended to be aggregated, or used for grouping or generating prompt pick lists.

In BI modeling, a characteristic of an entity which is descriptive rather than a unique identifier or an aggregative measure.

authentication

The process of validating the identity of a user or server.

authentication provider

The communication mechanism to an external authentication source. Functionalities, such as user authentication, group membership, and namespace searches, are made available through authentication providers.

calculated category

In Transformer, a category created based on a calculation. Calculated categories become part of the cube and can be applied to any measure.

calculated column

A column whose values are calculated from other columns, calculated columns, functions, and constants to derive new data for a model.

calculated measure

A measure whose values are calculated from other measures, calculated measures, functions, and numeric constants in an arithmetic equation.

calculated member

A member of a dimension whose measure values are not stored but are calculated at run time using an expression.

category

A set of items that are grouped according to a specific description or classification. Categories can be different levels of information within a dimension.

category code

In Transformer, a value that uniquely identifies every category within a dimension.

category count

In Transformer, a measure that records the number of unique, non-zero and non-missing values for the categories in the dimension and level specified by the user.

category set

In Transformer, a subset of the categories in a dimension, either from a single level or from different levels in the same dimension.

category viewer

In Transformer, the portion (right pane) of the Categories diagram that shows the category hierarchy of the selected dimension and provides a mechanism for manipulating the categories.

cloak

In Transformer, a customization option that restricts access to selected information in the cube. The cloak option removes a category and its descendants from a dimension, but summarizes the values in the ancestor categories. The category is not generated in any cube that uses the dimension view, and it is not viewable in any cube that uses the custom view. The category remains in the model. This option combines the features of summarize and suppress. You cannot cloak special categories.

conformed dimension

A dimension, with a single definition, that is reused or shared across multiple data subject areas. The common definitions for common dimensions allow data from different subject areas to be meaningfully compared and used in calculations and drill throughs from one area to another.

consolidation

The process of combining two or more duplicate records from a structural data source into a single record in the cube.

content locale

A code that is used to set the language or dialect used for browsers and report text, and the regional preferences, such as formats for time, date, money, money expressions, and time of day.

content store

The database that contains the data needed to operate, such as report specifications, published models, and security rights.

control cube

In Transformer, a cube that contains the structural information used to combine multiple time-segmented PowerCubes into a time-based partitioned cube.

convergence level

In Transformer, the level at which two or more alternate drill-down paths meet.

credentials

Information stored about the identity of an IBM Cognos user, usually a user name and password. You can assign your credentials to someone else so that they can use resources that you are authorized to use.

Credentials are created for IBM Cognos components. If a user schedules or programs an action, credentials must be stored in the content store.

cube

A multidimensional representation of data needed for online analytical processing, multidimensional reporting, or multidimensionl planning applications.

cube group

A set of similar cubes built by Transformer. Each cube group relates to a single level in one dimension of the model. Each member of the group is targeted at one of the categories in the level.

cube object

In Transformer, the object in a model that corresponds to a cube. The cube object specifies how to build the cube and how to reflect the status once the cube has been built.

currency record

Information defined in Transformer that is used in reporting components to display data in different currencies.

currency table

A table of information about currencies that the user maintains in Transformer or loads from an external data source.

custom view

A view that uses actions such as summarize, cloak, exclude, and apex to limit access to information in a cube to members of a given user class. User class views are replaced by custom views associated with the users, groups, or roles in a configured namespace.

data source

The source of data itself, such as a database or XML file, and the connection information necessary for accessing the data.

descendant category

In Transformer, a related category at a lower level in a dimension.

dimension

In Cognos Planning, a list of related items such as Profit and Loss items, months, products, customers, and cost centers, including calculations. The rows, columns, and pages of a cube are created from dimensions.

In Cognos BI, a broad grouping of descriptive data about a major aspect of a business, such as products, dates, or locations. Each dimension includes different levels of members in one or more hierarchies and an optional set of calculated members or special categories.

dimension line

In Transformer, a row of dimension names that appears either along the top of the dimension map window or just below the Measures and Data Sources option buttons in the Show Scope window.

dimension map

A table that shows the Transformer model in rows and columns. The columns in a dimension map are, for example, the dimensions, Dates, Products, and Sales Regions. The rows in a dimension map are the levels within the dimensions, for example, year, month, and day (for a time dimension) or continent, country, state, and city (for a regions dimension).

dimension view

In Transformer, a subset of a dimension used to create cubes that contain only selected aspects of the data represented by the complete model. For example, views based on specific countries or regions are a useful way of ensuring that users see only the data that is most relevant to them.

drill category

The immediate descendant (child) of a root category. A drill category is used only to define the properties of a drill-down path.

drill down

In a multidimensional representation of data, to access information by starting with a general category and moving downwards through the hierarchy of information. For example from Years to Quarters to Months.

drill through

To view the details linked to the data in a report, cube, or macro. For example, the user can drill through a value to view the detailed sales transactions for a particular customer. Any filtering of information in the original object is automatically applied.

A path used to view details linked to the data in a report, cube, or macro.

drill up

To navigate from one level of data to a less detailed level. The levels are set by the structure of the data.

duplicate record

A record with categories having identical non-measure values.

exclude

In Transformer, a customization option that restricts access to selected information in the cube. The exclude option omits a category and all data associated with the category and its descendants from a dimension view or custom view. The category and its descendants are not generated in any cube that uses the dimension view. The category remains in the model.

gateway

An extension of a Web server program that transfers information from the Web server to another server. Gateways are often CGI programs, but may follow other standards such as ISAPI and Apache modules.

hierarchy

The organization of a set of entities into a tree structure, with each entity (except the root) having one or more parent entities and an arbitrary number of child entities.

level

A set of entities that form one section of a hierarchy in a dimension and represent the same type of object. For example, a geographical dimension might contain levels for country, region, and city.

locale

A setting that identifies language or geography and determines formatting conventions such as collation, case conversion, character classification, the language of messages, date and time representation, and numeric representation.

locked dimension

In Transformer, a dimension to which new categories cannot be added. When processing data sources, any records which refer to values for this dimension that do not already exist as categories are ignored.

measure

A performance indicator that is quantifiable and used to determine how well a business is operating. For example, measures can be Revenue, Revenue/Employee, and Profit Margin percent.

measure folder

A folder that groups measures from a model into logical groupings. A measure folder can contain measures, and it can be a measure and have a value.

member

A unique item within a hierarchy. For example, Camping Equipment and 4 Man tent are members of the Products hierarchy.

member unique name

A path of member names, one from each level in a hierarchy, defining the exact location of the member from either an OLAP data source or a dimensionally modeled relational source. For example, Geography.Europe.France.Paris uniquely identifies Paris, France, distinguishing it from other instances of Paris in the City level.

model

A physical or business representation of the structure of the data from one or more data sources. A model describes data objects, structure, and grouping, as well as relationships and security. In Cognos BI, a model is created and maintained in Framework Manager. The model or a subset of the model must be published to the Cognos server as a package for users to create and run reports.

In Cognos Planning, a group of D-cubes, D-lists, D-links, and other objects stored in a library. A model may reside in one or more libraries, with a maximum of two for Contributor.

Model Definition Language

A proprietary language that can express the Transformer model definition. MDL is compatible with different versions of Transformer. Transformer model files that are formatted for export use the .mdl extension.

namespace

In XML and XQuery, a uniform resource identifier (URI) that provides a unique name to associate with the element, attribute, and type definitions in an XML schema or with the names of elements, attributes, types, functions, and errors in XQuery expressions.

object identifier

An identifier, which is usually a string of integers, that uniquely identifies a particular object within a distributed system.

orphan category

A temporary, manually-created category in a manual level that serves as a parent category for newly generated categories which have no position defined in the model.

package

A subset of a model, which can be the whole model, to be made available to the Cognos server.

performance indicator

See measure.

PowerCube

The type of cube created by IBM Cognos PowerPlay Transformer.

product locale

The code or setting that specifies which language, regional settings, or both to use for parts of the product interface, such as menu commands.

protected cube

A cube where members of different user classes have access to specific categories, measures, or dimensions, depending on their access privileges.

publish

In Cognos BI, to expose all or part of a Framework Manager model or Transformer PowerCube, through a package, to the Cognos server, so that the data can be used to create reports and other content.

In Cognos Planning, to copy the data from Contributor or Analyst to a data store, typically so that the data can be used for reporting purposes.

query

A request for information from a data source based on specific conditions: for example, a request for a list of all customers in a customer table whose balances are greater than $1000.

query item

A representation of a column of data in a data source. Query items may appear in a model or in a report and contain a reference to a database column, a reference to another query item, or a calculation.

query subject

A named collection of query items that are closely functionally related. Query subjects are defined using Framework Manager to represent relational data and form the set of available data for authoring reports in Query Studio and Report Studio. A query subject is similar to a relational view in that it can be treated as a table but does not necessarily reflect the data storage.

scope map

In Transformer, a color-coded dimension map that shows the association of dimensions and levels with each data source and measure in the model.

security object

An object such as a user, group, or role that is created for authentication and authorization purposes.

security provider

See authentication provider.

session

The time during which an authenticated user is logged on.

special category

A category that groups a set of regular categories from any level in the same dimension, without regard to their normal hierarchical organization. For example, in a dimension called Management that includes the levels Senior Management, Middle Management, and Junior Management, it is possible to have a special category called Social Committee that includes specific personnel from each of these levels.

structural data source

A data source that defines the structure of a Transformer model. It contains columns that map to levels and categories to build dimensions in the model.

subdimension

A tree of categories with levels that are independent of levels in the dimension. A subdimension can provide different details, or different levels of detail, for categories in a level. Subdimensions are permitted in dimensions with only one drill-down path or, in an alternate drill-down structure, at or below the convergence level.

summarize

In Transformer, a customization option that restricts access to selected information in the cube. The option sums all descendant categories for a category and suppresses the descendants.

summary

In reporting and analysis, an aggregate value that is calculated for all the values of a particular level or dimension. Examples of summaries include total, minimum, maximum, average, and count.

summary partition

A partition that contains pre-summarized values for the categories in higher levels of one or more dimensions. Information requests that can be satisfied from the summary partition use the pre-summarized values and, therefore, require less calculation at the time of the request.

suppress

In Transformer, a customization option that restricts access to selected information in the cube. The suppress option conceals a category within a dimension view. Within any cube that includes the view, the immediate descendants of the suppressed category link directly to its immediate ancestor category. In the cube, users see only the immediate ancestors and immediate descendants of the suppressed category.

time-based partitioned cube

A cube that combines multiple time-segmented PowerCubes based on the structural information in a control cube.

time-state measure

A measure that represents the state of a system at a specific point in time. For example, a time-state measure might show the number of employees in a company at the start of each month, or the product inventory on a given day for each week. A time-state measure can be aggregated by operations, such as totalling across other dimensions, but not across time.

transactional data source

A data source that contains records and that provides the measure values for cubes. In combination with one or more structural data sources, transactional data sources populate the model in Transformer. The separation of structural and transactional source data is a proven practice that improves data-processing time.

uniqueness

A designation for a level that indicates that each category in that level can be identified by its source value alone, without reference to its ancestors. The user must specify that the data is unique when a level is the convergence level for multiple drill-down paths or when the model contains multiple data sources. For example, employee IDs are unique source values, employee names are not.

user

Any individual, organization, process, device, program, protocol, or system that uses the services of a computing system.

user class

See custom view.

Index