## Appendix A. Accessibility features

- Accessibility features in Cognos Cube Designer .................................................. 133
- Keyboard shortcuts for Cognos Cube Designer ....................................................... 133

## Appendix B. Report considerations

- Calculated members in reports ................................................................. 137
- Relative time calculated members in reports .................................................. 139
- Removal of padding members from reports ..................................................... 140

## Appendix C. Troubleshooting

- Possible overflow in measure attributes ......................................................... 141
- In-memory aggregates fail to load ................................................................. 141
- Issues with dynamic cubes that contains members with duplicate level keys .......... 142

## Notices

- .................................................................................................................. 143

## Index

- .................................................................................................................. 147
Introduction

This document is intended for use with IBM® Cognos® Dynamic Cubes. It describes the processes required to model dimensional metadata and to create dynamic cubes to use as data sources in the Content Manager.

Audience

The following knowledge and experience can help you to use the product.

- Knowledge of OLAP concepts.
- Knowledge of your business requirements.
- An understanding of the structure of your data sources.
- Experience of installing and configuring applications.

Finding information

To find IBM Cognos product documentation on the web, including all translated documentation, access one of the IBM Cognos Information Centers (http://pic.dhe.ibm.com/infocenter/cogic/v1r0m0/index.jsp). Release Notes are published directly to Information Centers, and include links to the latest technotes and APARs.

You can also read PDF versions of the product release notes and installation guides directly from IBM Cognos product disks.

Accessibility features

Accessibility features help users who have a physical disability, such as restricted mobility or limited vision, to use information technology products. IBM Cognos Dynamic Cubes has accessibility features. For information about these features, see the accessibility section in this document.

IBM Cognos HTML documentation has accessibility features. PDF documents are supplemental and, as such, include no added accessibility features.

Forward-looking statements

This documentation describes the current functionality of the product. References to items that are not currently available may be included. No implication of any future availability should be inferred. Any such references are not a commitment, promise, or legal obligation to deliver any material, code, or functionality. The development, release, and timing of features or functionality remain at the sole discretion of IBM.

Samples disclaimer

The Sample Outdoors Company, Great Outdoors Company, GO Sales, any variation of the Sample Outdoors or Great Outdoors names, and Planning Sample depict fictitious business operations with sample data used to develop sample applications for IBM and IBM customers. These fictitious records include sample data for sales transactions, product distribution, finance, and human resources. Any resemblance to actual names, addresses, contact numbers, or transaction
values is coincidental. Other sample files may contain fictional data manually or machine generated, factual data compiled from academic or public sources, or data used with permission of the copyright holder, for use as sample data to develop sample applications. Product names referenced may be the trademarks of their respective owners. Unauthorized duplication is prohibited.
Chapter 1. What's new?

This section contains a list of new features for this release of IBM Cognos Dynamic Cubes.

For information about upgrading, see the *IBM Cognos Business Intelligence Installation and Configuration Guide*.

For information about new features for this release, see the *IBM Cognos Business Intelligence New Features Guide*.

To review an up-to-date list of environments that are supported by IBM Cognos Business Intelligence products, including information on operating systems, patches, browsers, web servers, directory servers, database servers, and application servers, see the [IBM Software Product Compatibility Reports (SPCR) page](www.ibm.com/support/docview.wss?uid=swg27037784).

### New features in 10.2.1.1

In IBM Cognos Dynamic Cubes 10.2.1.1, new features include dimension filters and measure dimension filters, measure folders, sorting measures, and embedded prompts and macros.

#### Dimension filters and measure dimension filters

You can now create dimension filters to restrict the members available in a published dynamic cube. For more information, see "Dimension filters" on page 58.

You can also create measure dimension filters to restrict the fact data available in a published dynamic cube. For more information, see "Measure dimension filters" on page 67.

#### Measure folders and sorting

You can now create folders in a measure dimension to contain regular measures and calculated measures. For more information, see "Creating a measure folder" on page 68.

You can also change the order in which measures and folders are sorted. For more information, see "Changing the sort order of measures and folders" on page 69.

#### Embedded prompts and macros

You can now embed prompts and macros in a calculated member or calculated measure expression. For more information about using prompts and macros, see the *IBM Cognos Framework Manager user Guide*.

### New features in 10.2.1

See the following topics for new features since the last release. Links to directly related topics are included.
Importing InfoSphere® Warehouse Cubing Services cube metadata

You can now import cube metadata from an IBM InfoSphere Warehouse Cubing Services model.

For more information, see "Import InfoSphere Warehouse Cubing Services cube metadata" on page 43.

Generating cubes and dimensions

From the Data Source Explorer in IBM Cognos Cube Designer, two new options are available to help reduce the overall time to build a cube. **Generate, Cube with dimensions using data sampling** creates a set of dimensions that are based on a selected fact table and the tables it joins. Each dimension is generated with one or more levels. **Generate, dimension using data sampling** creates a dimension with one or more levels that are based on the selected table.

For more information, see "Defining a dynamic cube based on a relational table" on page 62 and "Defining a dimension based on a relational table" on page 49.

The **Generate, Cube** option from the previous release has been renamed to **Generate, Cube with basic dimensions**. The functionality remains unchanged.

Aggregation rules

Three aggregation rules for measures have been added for this release. From the **Aggregation Rules** tab, you can access the **First**, **Last**, and **Current Period** options from the **Aggregation Rule** drop down list.

For more information, see "Aggregation rules" on page 29.

Aggregation Advisor

The Aggregation Advisor now suggests summary tables to assist in the loading of in-memory aggregates.

Improved security

Security features have been enhanced in the following areas for this release:

- Member security
  Security rules can now be stored in relational database lookup tables, better enabling the automation of security definitions for dynamic cubes.

- Dimension security
  It is now possible to secure user access to entire dimensions within a dynamic cube.

- Attribute security
  It is now possible to restrict user access to specific member attributes in a hierarchy. Member security definitions stored in database tables.

- Refresh security
  It is now possible to refresh security without having to restart a dynamic cube as long as there no significant changes to the modeled cube. If there are changes to dimensions, hierarchies, levels, or attributes, you must restart the dynamic cube.
Performance issues

In Cognos Cube Designer, there is a new Performance Issues tab that shows a list of all the performance issues for objects. These are issues that affect how well a dynamic cube performs when it is published and started.

For more information, see Validate a project and individual objects on page 45.

Centralized administration interface for dynamic cubes

A new page named Data Stores was added on the Status tab in IBM Cognos Administration. On this page, administrators can view, configure, manage, and monitor all dynamic cubes available in the IBM Cognos environment.

For more information, see Chapter 12, “Cognos dynamic cubes administration,” on page 113.
Chapter 2. Cognos Dynamic Cubes overview

In a dimensional data warehouse, you model relational database tables using a star or snowflake schema. This type of data warehouse differs from a traditional OLAP model in the following ways:

- It stores information about the data in fact and dimension tables rather than in proprietary OLAP data structures.
- It describes the relationships within the data using joins between the dimension and fact tables, the collection of dimension keys in a fact table, and the different attribute columns in a dimension table.

IBM Cognos Dynamic Cubes adds an in-memory relational OLAP component to the dynamic query mode server to provide a multidimensional view of a relational data warehouse with accelerated performance. You can then perform OLAP analysis using the Cognos Dynamic Cubes server.

Cognos Dynamic Cubes differs from Cognos dimensionally-modeled relational (DMR) data sources for the following reasons:

- It provides increased scalability and the ability to share data caches between users for better performance.
- It allows you to create a dynamic cube data source that is pre-loaded with dimensions.
- It allows for a richer set of dimensional modeling options and the explicit management of the member and data caches of a dynamic cube.

The benefits of Cognos Dynamic Cubes can be achieved only when using a dynamic cube as a data source. To use a dynamic cube as a data source, you must use the dynamic query mode.

Cognos Dynamic Cubes introduces a performance layer in the Cognos query stack to allow low-latency, high-performance OLAP analytics over large relational data warehouses. By using the power and scale of a relational database, Cognos Dynamic Cubes can provide OLAP analytics over terabytes of warehouse data.

Cognos Dynamic Cubes uses the database and data cache for scalability, and also uses a combination of caching, optimized aggregates (in-memory and in-database), and optimized SQL to achieve performance. The Cognos Dynamic Cubes solution includes the following characteristics:

- It uses simple, multi-pass SQL that is optimized for the relational database.
- It is able to minimize the movement of data between the relational database and the Cognos Dynamic Cubes engine.
  This data control is achieved by caching only the data that is required and by moving appropriate calculations and filtering operations to the database. At run time, only fact data is retrieved on demand.
- It is aggregate-aware, and able to identify and use both in-memory and in-database aggregates to achieve optimal performance.
  Aggregate awareness (aggregates tables that are created in the database and modeled into a dynamic cube) uses specialized log files to allow the dynamic query mode server to decompose queries to take advantage of the aggregate tables.
• It optimizes aggregates (in-memory and in-database) using workload-specific analysis.
  Aggregate Advisor, part of IBM Cognos Dynamic Query Analyzer, analyzes the performance of dynamic cubes using log files and provides suggestions for improving cube performance.
• It can achieve low latency over large data volumes, such as billions of rows or more of fact data and millions of members in a dimension.
  By using virtual cubes, companies can still present the complete view of the data, but need to refresh only smaller sets of data, leaving pre-cached query results for larger static sets. Users experience better performance for queries run against pre-cached results.

Evaluating your data

Before starting to model a cube, it is important to understand how your data affects the processing in IBM Cognos Cube Designer.

Referential integrity in data warehouses

Most databases today support referential integrity. However, it is typically turned off or is made declarative and instead is enforced during extract, transform, and load (ETL) processing. Erroneous modifications made to the data during or outside of the ETL process can create cases where a fact table has no matching dimension records.

Each data point in a dynamic cube is defined by a member from each dimension in the cube. If a value is required for some data point, then the SQL generated by Cognos Dynamic Cubes does not specify a filter on the table associated with a particular dimension if the member of that dimension is the All member. This allows for smaller SQL queries and also faster executing queries.

When a dimension is in scope, the join between the fact and dimension table is specified in the SQL query and the dimension is filtered by an explicit set of dimension key values. When the member of a dimension is the All member, dynamic cubes will not specify a filter for that dimension. All records are included, even records with invalid or missing dimension key values. This difference causes a discrepancy between values, depending upon which dimensions are involved in a query.

Even if your fact records have invalid or unknown dimension key values, you should validate your records before implementing Cognos Dynamic Cubes. Run an SQL query similar to the following for each dimension in a dynamic cube. This determines if there are any fact records with invalid dimension key values. Any returned data is the set of invalid dimension key values. If no data is returned, there are no referential integrity errors.

```sql
select distinct FACT.Key
from FactTable FACT
where not exists
(select *
from DimensionTable DIM
where DIM.Key = FACT.Key)
```

The SQL query can also be used as a subquery, to obtain the full set of records from the fact table.
If your fact table might contain records with invalid or unknown dimension key values, a common practice is to create a row in the dimension table to represent these dimension keys. New fact rows with invalid or unknown dimension key values can be assigned this dimension key value until the fact records and the dimension table can be updated with correct information. With this practice, records with problematic dimension key values are visible, regardless of which dimensions are involved in a report or analysis.

You should also validate snowflake dimensions.

You may have a situation where tables in a snowflake dimension are joined on a column for which the outer table did not contain values for rows in the inner table. In this case, the inner dimension table joins to the fact table, but the outer dimension table does not join to the inner dimension table.

To ensure that snowflake dimensions do not have this type of referential integrity error, run an SQL query similar to the following. In this example, the dimension is built from two tables, D1_outer and D2_inner. D2_inner is joined to the fact table. Key is the column on which the two dimension tables are joined.

```sql
select distinct INNER.Key
from D2_inner INNER
where not exists
  (select *
   from D1_outer OUTER
   where OUTER.Key = INNER.Key)
```
Chapter 3. Cognos Dynamic Cubes workflow

IBM Cognos Dynamic Cubes brings faster, more powerful, cube performance into the IBM Cognos reporting environment. Cognos Dynamic Cubes is used to improve access to large sets of data.

Figure 1 illustrates the relationship between the main activities performed using IBM Cognos Dynamic Cubes and the corresponding tools. IBM Cognos Cube Designer provides dynamic cube design and modeling capability. The Administration Console is used to deploy and manage the cube data. The dynamic query mode (DQM) server maintains the cube data. Studio applications use the data in reporting environments. In addition, various tools, such as Dynamic Query Analyzer, are used to analyze and optimize the data as necessary.

The following diagram shows the five major steps in a typical process flow, showing the users who are involved at each step.

Figure 1. Relationships between Cognos Dynamic Cubes activities and tools
Analyze the data

Before installing IBM Cognos Dynamic Cubes, the modeler and relational database administrator prepare for project implementation by completing the following tasks:

- Determining whether the data is a good candidate for Cognos Dynamic Cubes.
- Reviewing prerequisites to ensure correct implementation.

For more information about assessing your data and understanding prerequisites, see Chapter 2, “Cognos Dynamic Cubes overview,” on page 5.

Design and model a dynamic cube

The system analyst determines high-level business requirements and evaluates cube design against reporting requirements.

The modeler creates a basic dynamic cube, adds features to satisfy the business requirements and ensures that the cube is available to IBM Cognos Administration. Within IBM Cognos Cube Designer, the modeler performs tasks such as:

- Importing relational metadata to use as the basis for dynamic cube design.
- Designing dynamic, aggregate, and virtual cubes.
- Setting cube-level security for hierarchies and measures.
- Publishing the dynamic cube.

For more information about designing and modeling dynamic cubes, see the following topics:

- “Import metadata” on page 41
- “Model a dynamic cube” on page 61
- “Calculated members” on page 71
- Chapter 9, “Aggregate cube modeling,” on page 85
Deploy and manage a dynamic cube

After dynamic cubes are published to Content Manager, the Administrator handles the initial configuration and subsequent management. Within IBM Cognos Administration, the administrators perform tasks such as:

- Setting the Access Account property in the Administration Console.
- Assigning users, groups and roles to security views.
- Assigning a server group to the dispatcher.
- Assigning a routing set to all packages associated with a dynamic cube.
- Creating a routing rule to route queries for the routing set to the server group.
- Configuring the query service and the dynamic cube for a dispatcher.
- Starting the dynamic cube for initial use.
- Refreshing the dynamic cube, as necessary.
- Stopping the dynamic cube (soft or hard stop) while the data warehouse is being updated.
- Optionally, turning on logging. Log files are required to optimize the cube.
- Clearing workload logs.

For more information about deploying and managing dynamic cubes, see Chapter 12, “Cognos dynamic cubes administration,” on page 113 and the IBM Cognos Business Intelligence Administration and Security Guide.

Run reports using dynamic cube data

The report author uses the dynamic cube as a data source in reporting applications.

Optimize a dynamic cube

To optimize individual cube performance, the administrator can monitor the metrics of the dynamic cubes, and make changes, if necessary, to the cube configuration.

To further optimize performance, the system analyst can run a series of reports that are a representative workload against the dynamic cube. The resulting workload logs are used by Aggregate Advisor to return recommendations for additional in-memory and in-database aggregates. The analyst can also examine request execution log files in the Dynamic Query Analyzer. The log files help the analyst understand where time is spent within the dynamic cube engine, the type of SQL queries that are posed, how much time is spent executing the queries, and how many rows of data are returned. For information about Aggregate Advisor, see the IBM Cognos Dynamic Query Analyzer User Guide.

When you save in-memory aggregate recommendations to the content store, they are loaded automatically the next time the dynamic cube is started.
For in-database aggregate recommendations, the database administrator creates the aggregate tables in the database and the modeler uses IBM Cognos Cube Designer to model and publish the dynamic cube. For more information, see Chapter 9, “Aggregate cube modeling,” on page 85.

After new aggregates are published by the modeler, the administrator sets the in-memory aggregate size and restarts the dynamic cube to use new aggregates.

For detailed information, see Chapter 12, “Cognos dynamic cubes administration,” on page 113.

Workflow summary

To prepare for and manage project implementation, there are tasks external to the IBM Cognos software and tasks performed using IBM Cognos software. The following table shows a summary of responsibilities in each step of the workflow.

<table>
<thead>
<tr>
<th>Workflow</th>
<th>Responsibilities</th>
<th>Tools</th>
<th>Role</th>
</tr>
</thead>
<tbody>
<tr>
<td>Analyze, configure</td>
<td>Gather requirements and best practices.</td>
<td></td>
<td>Solutions architect</td>
</tr>
<tr>
<td></td>
<td>Determine best practices.</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Prepare an overall design.</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Perform hardware assessments.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Configure</td>
<td>Determine operating system administration changes.</td>
<td>O/S command tools, system administration console</td>
<td>System administrator</td>
</tr>
<tr>
<td></td>
<td>Perform middleware installation and maintenance.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Analyze, model</td>
<td>Design the database physical model.</td>
<td>Modeling tools, document/presentation software</td>
<td>Data architect</td>
</tr>
<tr>
<td></td>
<td>Design the multi-dimensional model.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Analyze, model</td>
<td>Gather business requirements.</td>
<td>Modeling tools, document/presentation software</td>
<td>Business/Application consultant</td>
</tr>
<tr>
<td></td>
<td>Design the logical model.</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Prepare the security definition.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Model, optimize</td>
<td>Design dynamic cubes.</td>
<td>IBM Cognos Cube Designer, IBM Cognos Dynamic Query Analyzer</td>
<td>Cognos modeler</td>
</tr>
<tr>
<td></td>
<td>Define security rules and views.</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 1. Workflow responsibilities by role
Table 1. Workflow responsibilities by role (continued)

<table>
<thead>
<tr>
<th>Workflow</th>
<th>Responsibilities</th>
<th>Tools</th>
<th>Role</th>
</tr>
</thead>
<tbody>
<tr>
<td>Manage, deploy</td>
<td>Configure and manage dynamic cubes.</td>
<td>Cognos Administration Console, Cognos Dynamic Query Analyzer</td>
<td>Cognos administrator (system)</td>
</tr>
<tr>
<td>Manage, deploy</td>
<td>Manage security of IBM Cognos objects, including dynamic cubes.</td>
<td>Cognos Administration Console</td>
<td>Cognos administrator (security)</td>
</tr>
<tr>
<td>Manage, deploy</td>
<td>Manage IBM Cognos data sources. Assign users to security views.</td>
<td>Cognos Administration Console</td>
<td>Cognos administrator (directory)</td>
</tr>
<tr>
<td>Optimize, Model</td>
<td>Evaluate overall performance. Run Aggregate Advisor.</td>
<td>Cognos Cube Designer, Cognos Dynamic Query Analyzer</td>
<td>Cognos administrator (system)</td>
</tr>
<tr>
<td>Run</td>
<td>Author reports, analyses or dashboards for use by collection of users</td>
<td>Cognos BI client applications</td>
<td>Cognos report author</td>
</tr>
</tbody>
</table>
| Configure, Model, Optimize | Implement database updates  
Perform database maintenance such as extract, transform and load (ETL) processes, backup and recovery. | Database administration console, ETL tools | Database administrator |
Chapter 4. Dimensional metadata and dynamic cubes

Understanding concepts relating to dimensional metadata and dynamic cubes helps you to plan and create effective dynamic cubes.

Dimensional metadata

In IBM Cognos Dynamic Cubes, dimensional metadata refers to dimensions and hierarchies. You can create commonly used dimensional metadata independent of any dynamic cubes in a project. The appropriate dimensional metadata can then be shared by one or more cubes in a project.

You can also create dimensional metadata that is connected to a specific dynamic cube.

Dimensions

In IBM Cognos Dynamic Cubes, you can create two types of dimensions: regular and parent-child.

A regular dimension is a collection of hierarchies and levels that describe one aspect of a measure, such as Customer or Product. This type of dimension can contain one or more hierarchies. A hierarchy uses levels to describe the relationship and order of dimension attributes. Related attributes and the joins that are required to group these attributes are defined in the dimension. For more information, see "Hierarchies."

A parent-child dimension contains dimension data based on a recursive relationship and is not level-based. This type of dimension can contain only a single parent-child hierarchy. For more information, see "Parent-child hierarchies" on page 20.

Data for regular dimensions and parent-child dimensions is typically stored in dimension tables.

Cognos Dynamic Cubes also supports degenerate dimensions. A degenerate dimension is a regular dimension for which dimension data is stored in a fact table. When modeling a dynamic cube based on a degenerate dimension, you do not need to specify a measure-to-dimension join.

Hierarchies

A hierarchy uses levels to describe the relationship and order of dimension attributes. For example, a Customer dimension might contain a Region hierarchy. For more information about attributes and levels, see "Attributes" on page 23 and "Levels" on page 22.

IBM Cognos Dynamic Cubes supports balanced, unbalanced, and ragged hierarchies. Padding members are used to balance unbalanced and ragged
hierarchies, so they appear as balanced hierarchies in the IBM Cognos studios. For more information, see “Padding members” on page 18.

**Multiple hierarchies**

Multiple hierarchies can be defined for dimensions containing level-based hierarchies.

You create multiple hierarchies for a dimension when you want to organize dimension members in different ways. For example, in a Time dimension, you can create hierarchies for Calendar year and Fiscal year.

Because dimension members in separate hierarchies can be used to represent the same entity, each hierarchy should contain the same lowest level members. For example, in a Time dimension, the Calendar hierarchy might have Year, Month, and Day levels. The Fiscal hierarchy might have Year, Quarter, and Day levels. The lowest level in both dimensions is the Day level.

Hierarchies that are modeled using a shared level can be optimized during query execution to remove non-intersecting values. To do this, you must ensure the Remove non-existent tuples property is set in a dynamic cube. For more information, see “Model a dynamic cube” on page 61.

**Balanced hierarchies**

In a balanced hierarchy, the branches of the hierarchy all descend to the same level. The parent of every member comes from the next highest level.

A balanced hierarchy can be used to represent time where the meaning and depth of each level, such as Year, Quarter, and Month, is consistent. They are consistent because each level represents the same type of information, and each level is logically equivalent. The following diagram shows an example of a balanced time hierarchy.

![Figure 3. Example of a balanced hierarchy](image)

**Unbalanced hierarchies**

Unbalanced hierarchies include levels that are logically equivalent, but each branch of the hierarchy can descend to a different level. In other words, an unbalanced hierarchy contains leaf members at more that one level. The parent of every member comes from the level immediately above.

An example of an unbalanced hierarchy is the following organization chart, which shows reporting relationships between employees in an organization. The levels
within the organizational structure are unbalanced, with some branches in the hierarchy having more levels than others.

![Unbalanced hierarchy diagram]

IBM Cognos Dynamic Cubes inserts padding members to balance such hierarchies. For more information, see “Padding members” on page 18.

**Ragged hierarchies**

In a ragged hierarchy, the parent of at least one member does not come from the level immediately above, but a level higher up.

The following diagram shows a Geography hierarchy with Continent, Region, State, and City levels defined. One branch has North America as the continent, Canada as the region, Manitoba as the state, and Winnipeg as the city. Another branch has Europe as the continent, Greece as the region, and Athens as the city, but has no entry for the state level because this level is not applicable. The parent of Athens is at the region level rather than the state level, creating a ragged hierarchy.

![Ragged hierarchy diagram]

*Figure 4. Example of an unbalanced hierarchy*

*Figure 5. Example of a ragged hierarchy*
IBM Cognos Dynamic Cubes inserts padding members to balance such hierarchies. For more information, see "Padding members."

Padding members

IBM Cognos Dynamic Cubes inserts padding members to balance unbalanced and ragged hierarchies. Padding members do not represent actual dimension members; they are visible only for navigational and performance reasons.

You can reference a padding member in an expression in the same way as any other hierarchy member.

Padding members can include a blank caption or the same caption as the parent. The following diagram illustrates a ragged hierarchy with a padding member included in the Europe branch. A blank caption was used as the caption for the padding member.

![Diagram of a ragged hierarchy with padding member](image.png)

*Figure 6. Example of a ragged hierarchy with blank padding member*

In the IBM Cognos studios, the metadata for this hierarchy with blank captions would show a level without a caption, as in the following example:

```
North America
├── Canada
│   ├── Manitoba
│       ├── Winnipeg

Europe
├── Greece
│   └── Athens
```

*Figure 7. Example metadata showing blank padding member*

The metadata for the same hierarchy using parent captions would show a level that uses the same caption as the parent, as in the following example:

```
North America
├── Canada
│   ├── Manitoba
│       └── Winnipeg

Europe
├── Greece
│   └── Athens
```

*Figure 7. Example metadata showing blank padding member*
A dimensional member can have only one child padding member.

The use of padding members can result in skewed calculations related to the members of a hierarchy level. For information about removing skewed data from reports, see “Removal of padding members from reports” on page 140.

 Extraneous padding members

With a level-based hierarchy, you can assign values from a dimension table to any member within the hierarchy; in other words to leaf members and non-leaf members. Data for non-leaf members can also be obtained by rolling up (aggregating) data from leaf members.

Tip: To roll up data for non-leaf members, the table that is used to model a level-based hierarchy must join to the fact table by using surrogate keys.

For example, a sales manager can also be a sales person with their own sales values. To assign sales values to the sales manager, the dimension table must contain a row in which the level key values for all levels below the manager level are Null.

For example, a sales manager can also be a sales person with their own sales values. The following example dimension table shows data for two sales people (Mark and Fred), and their sales manager (James). James is a non-leaf member which has a separate data value (100).

Table 2. Example dimension table

<table>
<thead>
<tr>
<th>Manager</th>
<th>Sales Person</th>
<th>Sales Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>James</td>
<td>Mark</td>
<td>15</td>
</tr>
<tr>
<td>James</td>
<td>Fred</td>
<td>20</td>
</tr>
<tr>
<td>James</td>
<td>&lt;null&gt;</td>
<td>100</td>
</tr>
</tbody>
</table>

Using IBM Cognos Dynamic Cubes, you can construct this hierarchy in one of the following ways:

• Create a path of extraneous padding members.

This option creates a full path of padding members from the non-leaf member to the leaf level to ensure that the hierarchy is balanced. It also provides a value at the lowest level so that the data can be rolled up. This is known as a rollup hierarchy.

The caption of these members can be blank or the same as the non-leaf member. If a non-leaf member has a value associated with it, this value is assigned to the padding member, which allows the contribution of a non-leaf member to its own rollup value.
• Remove the path of extraneous padding members.

Depending on the number of hierarchy levels, and the number of non-leaf member values, adding a path of extraneous padding members can result in a large hierarchy. To allow easier navigation of such a hierarchy, you can remove these paths.

To ensure that a hierarchy is balanced, you can remove a path of extraneous padding members only where a non-leaf member includes other leaf members. If paths are removed for any hierarchy, the entire dimension is identified as a non-rollup hierarchy. This prevents the query engine from assuming that the value of a parent is the rollup of its children. In addition, extraneous padding members are assigned a value of Null for all measures. This typically occurs when a detail filter is applied at a level below the lowest projected level in a report, or if the context filter (slicer) in a report contains multiple members from a single hierarchy.

The following example illustrates a hierarchy with a path of extraneous padding members.

Table 3. Example hierarchy showing extraneous padding member

<table>
<thead>
<tr>
<th>Manager</th>
<th>Sales Person</th>
<th>Sales Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>James</td>
<td>Mark</td>
<td>15</td>
</tr>
<tr>
<td>James</td>
<td>Fred</td>
<td>20</td>
</tr>
<tr>
<td>James</td>
<td>James</td>
<td>100</td>
</tr>
</tbody>
</table>

By default, the path of extraneous padding members is removed in a level-based hierarchy. To show or remove the path, you must set the Show Extraneous Padding Members property. For more information about setting this property, see “Model hierarchies” on page 50.

Parent-child hierarchies

A parent-child hierarchy contains relational dimension tables based on a recursive relationship for which there are no predefined levels. For example, an Employee parent-child hierarchy may specify Supervisor as the parent member, and Employee as the child member. The relationships within the data determine what is visible to report users in the IBM Cognos studios, and you can drill down from member to member according to the defined relationships.

IBM Cognos Dynamic Cubes supports parent-child hierarchies.

Data members

With a parent-child hierarchy, you can assign values from a dimension table to any member within the hierarchy; in other words to leaf members and non-leaf members. Data for non-leaf members can also be obtained by rolling up (aggregating) data from leaf members.

For example, a sales manager can also be a sales person with their own sales values. The following example dimension table shows data for two sales people (Mark and Fred), and their sales manager (James). In this example, Mark and Fred are leaf members, and James is a non-leaf member.
Table 4. Example dimension table

<table>
<thead>
<tr>
<th>Sales person</th>
<th>Sales</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mark</td>
<td>15</td>
</tr>
<tr>
<td>Fred</td>
<td>20</td>
</tr>
<tr>
<td>James</td>
<td>100</td>
</tr>
</tbody>
</table>

In the corresponding hierarchy structure, sales person values roll up to the sales manager. This is known as a rollup hierarchy.

The following example illustrates the report data for a rollup hierarchy with non-leaf members shown. The report includes two values for non-leaf member James - the child value that is assigned from the dimension table (100) and the total rolled up sales value that includes this child value (135).

Table 5. Example report data with a non-leaf member shown

<table>
<thead>
<tr>
<th>Sales person</th>
<th>Sales</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mark</td>
<td>15</td>
</tr>
<tr>
<td>Fred</td>
<td>20</td>
</tr>
<tr>
<td>James</td>
<td>100</td>
</tr>
<tr>
<td>James</td>
<td>135</td>
</tr>
</tbody>
</table>

The following example illustrates the same report data using a non-rollup hierarchy, where non-leaf members hidden.

Table 6. Example report data with a non-leaf member hidden

<table>
<thead>
<tr>
<th>Sales person</th>
<th>Sales</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mark</td>
<td>15</td>
</tr>
<tr>
<td>Fred</td>
<td>20</td>
</tr>
<tr>
<td>James</td>
<td>135</td>
</tr>
</tbody>
</table>

Rolling up report data in a non-rollup hierarchy causes two problems:
- Data for non-leaf members is not explicitly shown because it is already rolled up.
  To work out the individual value of a non-leaf member, you must extrapolate the data.
- If a parent-child hierarchy contains hidden non-leaf members, the entire dimension is identified as a non-rollup hierarchy.
  This prevents the query engine from assuming that the value of a parent is the rollup of its children. You must set the data members to be visible to allow a hierarchy to be identified as a rollup hierarchy.

When you model a dynamic cube, it is important to consider the presentation of a hierarchy against the effect on reports/analyses posed against the hierarchy, the parent dimension, and related hierarchies.

By default, non-leaf members are hidden in a parent-child hierarchy. To show or hide non-leaf members, you must set the Show Data Members property. For more information about setting this property, see "Model parent-child hierarchies" on page 55.
If the **Show Data Members** property is set to true, a child member is added to each non-leaf member in a parent-child hierarchy. The caption of these members can be blank or the same as the non-leaf member. If a non-leaf member has a value associated with it, this value is assigned to the child data member, which allows the contribution of a non-leaf member to its own rollup value.

**Levels**

A level is a collection of attributes related to one aspect of a hierarchy. For example, a Region hierarchy can contain States and City levels.

For more information about attributes, see “Attributes” on page 23.

You can define an All level at the highest level of a hierarchy. An All level contains a single member that aggregates data from all members in the child levels of the hierarchy. For example, you can include an All level in a Region hierarchy that aggregates data for all cities, in all states, in all regions.

**Important:** There are many ways to model a hierarchy using levels. Whether you follow best practice or different modeling techniques, it is important that you define each level so that the level key attributes uniquely identify the values in that level.

**Best practice modeling**

Both star and snowflake schemas can be used to implement best practice modeling. For example, in a star schema the relational data for each dimension is stored in a single dimension table that contains ID columns for each of the levels in the dimension, and each ID column uniquely identifies the values in the level. You might have a single dimension table for the Region dimension that contains the following columns:

<table>
<thead>
<tr>
<th>Columns in a best practice Region dimension table</th>
</tr>
</thead>
<tbody>
<tr>
<td>City ID (Primary key)</td>
</tr>
<tr>
<td>City name</td>
</tr>
<tr>
<td>City mayor</td>
</tr>
<tr>
<td>State ID</td>
</tr>
<tr>
<td>State name</td>
</tr>
<tr>
<td>State governor</td>
</tr>
<tr>
<td>Region ID</td>
</tr>
<tr>
<td>Region name</td>
</tr>
</tbody>
</table>

**Alternative modeling**

If you do not have unique ID data columns for each level in your hierarchy, you must be careful when you define the level key attributes for each level. For example, you might have a single dimension table for the Region dimension that contains the following columns:
Table 8. Example of a single dimension table using alternative modeling

<table>
<thead>
<tr>
<th>Columns in an alternative Region dimension table</th>
</tr>
</thead>
<tbody>
<tr>
<td>City ID (Primary key)</td>
</tr>
<tr>
<td>City name</td>
</tr>
<tr>
<td>City mayor</td>
</tr>
<tr>
<td>State name</td>
</tr>
<tr>
<td>State governor</td>
</tr>
<tr>
<td>Region name</td>
</tr>
</tbody>
</table>

You can create a hierarchy that contains Region, State, and City levels, like in the best practice modeling example. However, you must carefully define the level key attributes to ensure that each row in the level can be uniquely defined. For example, City name does not uniquely define the City level because there are cities with the same name in the United States and in England. The only way to uniquely define the City level is with the combination of the Region name, State name, and City name attributes, as shown in the following table:

Table 9. Example of unique level key attributes using multiple columns

<table>
<thead>
<tr>
<th>Level</th>
<th>Level key attributes</th>
<th>Level related attributes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Region</td>
<td>Region name</td>
<td></td>
</tr>
<tr>
<td>State</td>
<td>Region name, State name</td>
<td>State governor</td>
</tr>
<tr>
<td>City</td>
<td>Region name, State name, City name</td>
<td>City mayor</td>
</tr>
</tbody>
</table>

**Joins**

A join combines columns from two relational tables using an operator to compare the columns. A join uses attributes that reference columns in the tables being joined.

The simplest form of a join uses two attributes: one that maps to a column in the first table and one that maps to a column in the second table. You also specify an operator to indicate how the columns are compared. For example, “Time ID = time_id”.

A join can also model composite joins where two or more columns from the first table are joined to the same number of columns in the second table. A composite join uses pairs of attributes to map corresponding columns together. Each pair of attributes has an operator that indicates how that pair of columns is compared. For example, “Customer Number = customer_number AND Store Number = store_number”.

A join also has a type and cardinality. The join types map to relational join types. Joins are primarily used to join the cube dimensions to the relational tables. Joins can also be used to join dimension tables together in snowflake schema.

The most common type of join is the one-to-many equality join.

**Attributes**

An attribute is an item used to describe part of a level. For example, a Product level can have a Color attribute. An attribute contains an expression that can be a
simple mapping to a data source column or a more complex expression. Complex expressions can combine multiple columns or attributes. They can use functions that are supported against a relational data source, including user-defined functions, if necessary.

When modeling levels in IBM Cognos Cube Designer, there are some special attributes you can define:

- **Member caption** does not appear as a separate attribute of a level; it is used only as the caption for hierarchy members.
- **Member description** appears as a separate attribute with the name `level name description`.
- **Level unique key** appears as a separate attribute with the name `level name key`.

When additional attributes are used in an expression, they cannot form attribute reference loops. For example, if Attribute A references Attribute B, then Attribute B cannot reference Attribute A.

Attribute names must be unique from the names of all other attributes in a dimension.

---

**Dynamic cubes**

A dynamic cube represents a dimensional view of a star or snowflake schema. It is based on a single fact table and defines the relationships between dimensions and measures.

To model a basic dynamic cube, you must ensure that it contains the following items:

- A measure dimension that contains at least one measure
- At least one dimension
- At least one hierarchy and associated levels defined for each dimension
- Mappings between the measures and dimensions
- Attributes that reference table columns either directly, by expressions, or by an expression that is a constant value

Measures are used to aggregate data from a fact table using specified dimensions. They describe data calculations using columns in a relational table. The following diagram shows how measures relate to relational data.
Dimensions are connected to a measure using joins. A hierarchy provides a way to calculate and navigate a dimension. It stores information about how the levels within a dimension are related to each other, and how they are structured. Each dimension has one or more hierarchies that contain levels with sets of related attributes. The following diagram shows how dimensions are built from relational tables.

Figure 9. Relationship between measures and relational data
In a star schema, joins are used to connect tables to create a dimension or a measure. Joins can also connect a measure dimension to specific dimensions. The dimensions reference their corresponding hierarchies, levels, attributes, and related joins. A measure dimension references its measures, attributes, and related joins. In a snowflake schema, joins can also connect tables between dimensions. The following diagram shows how the items fit together in a dynamic cube and map to a relational snowflake schema.

Figure 10. Relationship between dimensions in a project and the source relational tables
Measures

In IBM Cognos Dynamic Cubes, you can define regular measures and calculated measures.

Regular measures are mapped directly to a database column of numerical data or defined by an expression. If defined by an expression, the expression is constructed from relational metadata and cannot include dimensional constructs and functions.

Calculated measures are computed in the context of a dynamic cube and computed in the dynamic query server. The expression is constructed from cube metadata and uses dimensional constructs and functions. Dimensional expressions are required when it is necessary to traverse hierarchical relationships or compute complex calculations which are difficult or impossible with relational expressions. With dimensional expressions, you have the ability to access parent/child relationships, to calculate parallel periods, to use set operations and to define expressions which are evaluated based on its context within a query.
There are some behavior similarities between calculated measures and calculated members. For information about calculated members, see “Calculated members” on page 71.

In Cognos Dynamic Cubes, a measure dimension, containing a set of measures, is used in a dynamic cube as the center of a star schema. The physical grouping of measures into a single fact table implies that they share one area of interest. Each measure references the attributes that are used in measure-to-dimension joins. Each measure also references the attributes and joins that are used to map the additional measures across multiple database tables. The value of a measure is meaningful only within the context of the dimensions in a cube. For example, a revenue of 300 has no meaning on its own, but does have meaning in the context of dimensions, such as Region and Time. For example, the revenue for New York in January is 300. Common examples of measures are Revenue, Cost, and Profit.

Simple arithmetic expressions can often be evaluated either by the relational database or in the context of the cube. If a measure expression can be evaluated in either context, it may be preferable to choose a relational expression. Relational databases usually have access to a wider range of functions and may be more efficient. If a database is constrained in terms of resources, an alternative is to use calculated measures.

**Regular aggregates**

Each measure has a regular aggregate. Aggregation rules can be used in addition to the regular aggregate. Aggregation rules define how a measure is aggregated in relation to one or more dimensions. A measure is aggregated by first applying the regular aggregate to all dimensions not specified by aggregation rules, then applying aggregation rules in the order they are listed.

A semi-aggregate measure is a measure that may aggregate differently relative to one or more dimensions within a cube. For example, relative to warehouses, inventory levels are additive. Relative to time, inventory levels are computed as of a point in time. This is typically the first or last occurrence within a time period (first or last day of the month). Therefore, an inventory level measure would have a Regular Aggregate of Sum, and an Aggregation Rule of First or Last relative to the Time dimension.

The **Regular Aggregate** property can have values of Average, Calculated, Count, Count Distinct, Count Non Zero, Custom, Maximum, Median, Minimum, Standard Deviation, Sum or Variance.

The Custom value indicates that the value of the measure is computed by an external business process. Custom measures are a specialized form of non-distributive measure that do not roll up. Values must exist in the measure or aggregate tables at the precise level of aggregation required for a query, otherwise the values are shown as Null. You can customize measure values using advanced business logic and make those values available in IBM Cognos Business Intelligence.

The Calculated value controls the order of operations for calculations. When you use a Calculated Regular Aggregate, IBM Cognos Dynamic Cubes first aggregates each measure in the expression using its Regular Aggregate property. Then, it uses the values of the aggregated measures to calculate the expression.
Use Sum and Count aggregates rather than Average where possible. You can also use simple calculations by selecting a measure and assigning a rule, such as Average.

Table 10. Sample data for calculated Regular Aggregate example

<table>
<thead>
<tr>
<th>Location</th>
<th>Time</th>
<th>Sales</th>
<th>Average Returns</th>
</tr>
</thead>
<tbody>
<tr>
<td>USA</td>
<td>Q1</td>
<td>10</td>
<td>2</td>
</tr>
<tr>
<td>USA</td>
<td>Q2</td>
<td>30</td>
<td>4</td>
</tr>
<tr>
<td>USA</td>
<td>Q3</td>
<td>50</td>
<td>6</td>
</tr>
</tbody>
</table>

Sales is defined with a Regular Aggregate of Sum. Average Returns are defined with a Regular Aggregate of Average.

In this example, the calculated measure, Measure A, is defined by the expression (Sales - Average Returns).

If Measure A is assigned a Regular Aggregate value of Sum, its value is computed as follows if grouping by distinct values of Location.

\[
\begin{align*}
10 - 2 &= 8 \\
30 - 4 &= 26 \\
50 - 6 &= 44 \\
\hline \\
\text{Measure A} &= 8 + 26 + 44 = 78
\end{align*}
\]

If Measure A is assigned a Regular Aggregate value of Calculated, its value is computed as follows if grouping by distinct values of Location.

Sales \(= 10 + 30 + 50 = 90\) \\
Average Returns \(= \frac{2 + 4 + 6}{3} = 4\) \\
\[
\begin{align*}
\text{Measure A} &= 90 - 4 = 86
\end{align*}
\]

**Aggregation rules**

Each measure has a regular aggregate. Aggregation rules can be used in addition to the regular aggregate. Aggregation rules define how a measure is aggregated in relation to one or more dimensions. A measure is aggregated by first applying the regular aggregate to all dimensions not specified by aggregation rules, then applying aggregation rules in the order they are listed.

Aggregation rules can be

- Distributive (Count, Sum, Maximum, Minimum)
- Non-distributive (Average, Standard Deviation, Variance)
- Time state (First, Last, Current Period)

Distributive measures can be aggregated from one level to the next. Existing aggregate values can be used to compute higher level aggregates. Non-distributive measures must be calculated from the base fact table data. They cannot be aggregated from one level to the next.

**Non-distributive measures**

Non-distributive measures must always be aggregated from the detail fact table grain and cannot be aggregated from one summary level to the next.
A non-distributive measure is a measure that is defined with a non-distributive aggregation rule such as:

- Count Distinct
- Average
- Standard Deviation
- Variance

Aggregate tables can be used only if they are calculated from the exact group of levels of the SQL query. If none of the aggregate tables exactly match the required roll ups, the aggregate value must be computed from the fact table. As a result, higher level aggregations of non-distributive measures against a large fact table can take longer to calculate than measures that can take advantage of external aggregate tables.

A dynamic cube stores the values of non-distributive measures in its data cache for later use.

When calculating summary values in a query, non-distributive measures require a separate SQL query for each summary. These summary values are query-specific and are not stored in the data cache.

For a cross tab report with row/column summaries, each summary requires a separate SQL query which, depending upon the underlying database, can have an impact on query performance.

Unlike non-distributive measures, distributive measures can always be aggregated from one level to the next. For example, the sum of Sales for a quarter can be calculated by summing monthly sales data.

**Time state aggregation rules**

The **First**, **Last** and **Current Period** aggregation rules represents the state of a measure at specific times. They are commonly used in inventory or account balances. There are several points to consider when using time state aggregation rules:

- Time state aggregation rules are computed at the granularity of the fact table. If the fact table is at a lower level of granularity than the dimension associated with the measure with a **First**, **Last**, or **Current Period** aggregation rule, IBM Cognos Cube Designer issues a warning.
- Aggregation cannot be computed correctly across multiple cubes. If a virtual cube contains a measure in which the underlying base measures have aggregation rules, Cognos Cube Designer issues a warning. The warning is issued only if the base cubes containing the aggregation rules exists in the project model.
- If there is no value associated with the appropriate leaf level member of the aggregation rule, the value of the measure is NULL.
- Time state aggregation rules are not affected by member security.
- Time state aggregation rules are not affected by attribute security.
- Time state aggregation rules are not supported for parent child hierarchies.
- If the dimension upon which the time state aggregation rule is based is secured for a user, the value for the measure is computed as that for the default member of the dimension as per the rules established for dimension security.
Errors must be corrected to be able to publish the cube. Warnings are informational, and do not prevent the cube from being published.

**First**

The **First** aggregation rule provides the measure value associated with the first leaf level descendant of the current member of the dimension for which the semi-aggregation rule is defined. For example, a time hierarchy contains years, quarters, and months, and you are examining data at the quarter level. For each quarter, the First rule reports the measure value from the first month of the quarter. When you examine data at the year level, the rule reports the first value from the first month in the first quarter of each year.

**Last**

The **Last** aggregation rule provides the measure value associated with the last leaf level descendant of the current member of the dimension for which the semi-aggregation rule is defined. For example, if a time hierarchy contains years, quarters, and months, and you are examining data at the quarter level, for each quarter, the Last Period rule reports the measure value from the last month of each quarter. When you examine data at the year level, it reports the value from the last month in the last quarter of each year.

**Current Period**

The **Current Period** aggregation rule provides the measure value associated with the leaf level descendant of the current member of the Time dimension that corresponds to the Current Period relative time member. If the current period is not a descendant of the current member, it provides the value of the Last leaf level descendant. 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.

The **Current Period** aggregation is only supported when it is defined relative to a dimension identified as a time dimension. The associated dimension must be a time dimension and each of the hierarchies in the time dimension must have the relative time property enabled.

Relative time and security cannot be enabled on a hierarchy at the same time. Therefore, **Current Period** is not supported on a secured time hierarchy.

**Time state aggregation rules with multi-hierarchy dimensions**

For a measure with a time state aggregation rule relative to a multi-hierarchy dimension, tuple values are computed according to the following rules:

**Rule 1:**

If a tuple has a non-ALL member from any hierarchy of a multi-hierarchy dimension, only the non-ALL members are resolved to the corresponding leaf level member for the time state aggregation rule.
For example, the Time dimension has two hierarchies Time.Actual and Time.Fiscal. Both have ALL members. The Closing Inventory measure has aggregation rule of Last.

The tuple \((\text{Closing Inventory, Time.Actual.ALL, Time.Fiscal.2012})\) resolves to \((\text{Closing Inventory, Time.Actual.ALL, Time.Fiscal.2013Jan})\). The result is Closing Inventory for 2013Jan because the aggregate rule is Last and 2013Jan is the last month of fiscal year 2012.


Rule 2

If a tuple only projects ALL members from a multi-hierarchy dimension, only the default member of default hierarchy is resolved for the time state aggregation rule.

For example, hierarchies Time.Actual and Time.Fiscal, both have ALL members. Time.Actual.ALL is the default member of default hierarchy.


---

**Virtual cubes**

In IBM Cognos Dynamic Cubes, a virtual cube consists of two merged cubes. You can merge cubes by using the following combinations:

- Merge two source cubes.
- Merge two virtual cubes.
- Merge one source cube with one virtual cube.

By combining two virtual cubes, or one source cube with a virtual cube, you can merge more than two cubes into a single virtual cube.

Some advantages of using virtual cubes include the following points:

- Virtual cubes use less memory than physical cubes.
- There is reduced cube refresh latency.
- You can add volatile information to a lookup cube.
- You can join cubes to present consolidated data and provide more sophisticated calculations.
- Each source cube can be derived from a separate data source.

A virtual cube must contain the following objects:

- A virtual measure dimension that contains one or more virtual measures.
- At least one dimension that contains one or more virtual hierarchies.

It can also contain virtual calculated measures, and virtual calculated members.

When you create a virtual cube, the following objects are added, if they exist in at least one source cube:
Virtual dimensions and hierarchies

Any dimensions and hierarchies with identical names in the source cubes are known as conformed dimensions and conformed hierarchies. These objects are added to the virtual cube as merged virtual dimensions and virtual hierarchies.

For example, two source cubes with a Time dimension are merged into a virtual dimension also named Time.

Any dimensions and hierarchies that do not have identical names, or that exist in only one of the source cubes, are known as non-conformed dimensions and non-conformed hierarchies. These objects are added to the virtual cube as new virtual dimensions and virtual hierarchies.

For example, if source cube 1 contains a Sales Q3 hierarchy, and source cube 2 contains a Sales Q4 hierarchy, the dimensions are not merged because the names do not match. Instead, two virtual hierarchies, Sales Q3 and Sales Q4, are added to the virtual cube.

If a virtual cube contains a non-conformed hierarchy, the virtual cube queries both source cubes to retrieve data only if one of the following conditions is met:

• The non-conformed hierarchy is deleted from the virtual cube.
• The virtual hierarchy includes an All member and the query includes this member.
  This can occur if the All member is referenced explicitly in the query or if the All member is the default member.

If neither of these conditions is met, the virtual cube queries only the source cube with the non-conformed hierarchy, and never the second source cube.

Virtual measures

Any measures with identical names in the source cubes are added to the virtual cube as merged virtual measures. Any measures that do not have identical names, or that exist in only one of the source cubes, are added to the virtual cube as new virtual measures.

Important: It is possible to merge measures only when the regular aggregate is one of the following: Sum, Maximum, Minimum, or Count. It is not possible to merge non-distributive measures or a distributive measure with an aggregation rule applied.

When merging measures from two source cubes, if there is a conflict between the data format of each measure, the data format of the merged virtual measure is set to * or unknown. For example, if a measure in source cube 1 has a US currency data format, and a measure in source cube 2 has a UK currency data format, the data format cannot be merged.
**Virtual levels**

Source cubes containing identical levels in a hierarchy (same number of levels and identical names), are merged as virtual levels. If levels in the source cubes are not identical, level names from the first source cube are used as the names of the virtual levels. If one source cube contains more hierarchy levels than the second source cube, the extra levels are added as the lowest levels of the virtual hierarchy.

For example, source cube 1 contains a Time hierarchy with Year, Quarter, and Month levels. Source cube 2 also has a Time hierarchy with Year, Month, Day, and Time levels. When they are merged, a Time virtual hierarchy is created with Year, Quarter, and Month, and Time virtual levels with the following members:

- The Quarter virtual level contains Quarter members from source cube 1 and Month members from source cube 2.
- The Month virtual level contains Month members from source cube 1 and Day members of source cube 2.
- The Time virtual level contains Time members from source cube 2.

**Virtual members**

For a virtual hierarchy that is merged from two conformed dimensions, all hierarchy members from the source cubes are available as virtual members. If the level key for each source member is identical, members are added to the virtual cube as merged virtual members. Any members that do not have matching level keys are added to the virtual cube as new virtual members.

**Tip:** To browse virtual members, ensure that each source cube is deployed as data source to the content store and started.

**Calculated measures and calculated members**

Calculated measures and calculated members from source cubes are not added to a virtual cube. To use calculated measures or members from source cubes, you must manually define them in the virtual cube.

For more information, see "Calculated members" on page 71.

**Aggregate cubes**

Aggregate cubes are unavailable in a virtual cube because a virtual cube can retrieve data only from source cubes, not by querying a data source.

**Support for multiple locales**

If source cubes include support for multiple locales, a virtual cube also has multiple locale support.

A virtual cube automatically supports all locales defined in the source cubes. For example, in source cube 1, English and French are defined as supported locales. In source cube 2, English and Japanese are defined as supported locales. In the virtual cube, English, French and Japanese are included as supported locales.

A virtual cube also supports the use of multilingual names and captions for a virtual cube, virtual dimensions, virtual hierarchies, virtual levels, and virtual measures. However, with the exception of the All member caption, multilingual
names and captions from source cubes are not automatically added to a virtual cube. To use multilingual names and captions from source cubes, you must manually define them in the virtual cube.

**Manual merging of source objects**

It is possible to manually merge objects in a virtual cube that could not be merged automatically. For example, source cube 1 contains a Time dimension and source cube 2 contains a Fiscal Time dimension. They are not merged, so two virtual dimensions Time and Fiscal Time are added to the virtual cube. If both dimensions contain the same structure and data, you could manually merge them into one virtual dimension named Time. You could then delete the redundant Fiscal Time virtual dimension.

You cannot reference a source object more than once in a virtual cube. For example, if the Time source hierarchy is used in the Time virtual hierarchy, it cannot also be used in the Fiscal Time virtual dimension.

**Virtual cube scenarios**

Common scenarios for using virtual cubes are described here. You can combine these scenarios based on your specific needs.

**Cubes with partitioned data**

Sales information for a large region is stored in two cubes. Fact data for each cube can originate from a single fact table or two separate fact tables. One cube, WestSales, stores sales information for the west region, and the other cube, EastSales, stores sales information for the east region. WestSales and EastSales have the same structure. To provide a combined view of the sales data, you can define a virtual cube called AllSales to merge the two regional cubes.

**Cubes with pre-cached historical data and current data**

Sales information is stored in a single cube called AllSales. The cache of this large cube must be rebuilt frequently to reflect the updates in the database. The rebuilding process usually takes a long time.

To address this problem, you can split AllSales into two cubes: one to record the historical sales information (HistoricSales), and another to record the daily sales information for the current month (CurrentMonthSales). You can then define a virtual cube called VirtualSales to join these two cubes. By reorganizing the cubes this way, performance is improved in the following ways:

- Because you refresh data only for CurrentMonthSales, cube refreshing performance is improved.
- Because query results from HistoricSales are pre-cached, and CurrentMonthSales is small in size, performance for queries run against the sales data of the entire time period is improved.
- Because of the smaller size of CurrentMonthSales, performance for queries run against the sales data of the current month is improved.
Cubes with shared dimensions

Sales information is stored in a single cube called GlobalSales. You need to convert some sales figures into other currencies. You could add exchange rates to this cube, but the cube might contain redundant data and would be hard to maintain.

Instead, you can create a cube called ExchangeCurrency to store the exchange rates, and define a virtual cube SalesConversion to perform currency conversion for the sales data. GlobalSales and ExchangeCurrency share some dimensions but do not have the same structure.

Aggregate cubes

In IBM Cognos Cube Designer, you can model aggregate cubes within a dynamic cube when the imported data source for a dynamic cube contains fact tables with pre-aggregated data.

IBM Cognos Dynamic Cubes supports the use of aggregate cubes created in a dynamic cube and rewrites queries to use the underlying aggregate tables whenever possible. For information about aggregate cube modeling, see Chapter 9, “Aggregate cube modeling,” on page 85.

Aggregate tables

Although it is a best practice to store the lowest level of data in a detail fact table in a data warehouse, selected data can be summarized in a separate table known as an aggregate table. An aggregate table contains detail fact data that is aggregated at a higher level relative to one or more of the dimensions associated with the data.

Using aggregates is critical to achieving performance over large scales for the following reasons:

- It allows you to use pre-calculated data from a data warehouse.
- It decreases the amount of data required to be accessed from the data warehouse.

Some database vendors use special table types for aggregate tables. For example, IBM DB2® uses Materialized Query Tables (MQTs) and Oracle uses Materialized Views. The relational database understands that these special tables are aggregates and routes to them for performance if the database can determine they are applicable and faster. The aggregate awareness feature in Cognos Dynamic Cubes can also use these tables so that a dynamic cube routes to these aggregates tables rather than relying on the database to do the routing.

To increase performance, more than one aggregate table can be necessary in a schema. However, if an aggregate table summarizes data at too high a level within one or more hierarchies, the aggregates can be applicable to only a few queries. In addition, if many dimensions are used, it can be difficult to design frequently used aggregate tables.

When creating aggregate tables, refer to the documentation for your database for information about creating a data warehouse, in particular indexing your data, and co-locating fact and dimension tables. Cognos Dynamic Cubes supports these concepts:
• Sharing common dimension tables if fact and aggregate tables are co-located in the same storage space.
• Use of separate dimension tables for aggregate tables (co-locating dimension and fact data).
• Inclusion of dimension level keys entirely within an aggregate table to avoid joins to dimension tables.
• Partitioning of data.

**In-database aggregates**

In-database aggregates are aggregate tables that a database administrator can create and apply to the database. After the database has been updated, a modeler must model an aggregate cube for each created aggregate table in the database and redeploy the dynamic cube to the content store.

**In-memory aggregates**

In-memory aggregates are aggregate tables that can be applied by the IBM Cognos Business Intelligence server the next time the cube is started. These aggregates are stored in the content store.

**Aggregate Advisor**

Aggregate Advisor is an external tool, available with IBM Cognos Dynamic Query Analyzer, that can analyze the underlying model in a dynamic cube data source and recommend which aggregates to create. These aggregates can be created both in-database and in-memory.

Aggregate Advisor can also reference a workload log file that allows it to suggest aggregate tables (in-database or in-memory) that correspond directly to the reports contained in the log file.

The Aggregate Advisor does not include recommendations for the following types of measures:

• Calculated measures
  The Aggregate Advisor recommends aggregates to accelerate queries that are processed by the underlying database. Because calculated measure expressions are processed in the dynamic query engine, there are no corresponding aggregate recommendations for these types of expressions.

• Semi-aggregate measures
  Semi-aggregate measures are not supported by the aggregate cache. However, you can model an aggregate cube to an existing in-database aggregate with a semi-aggregate measure. If there is an exact match between a query and an aggregate cube with a semi-aggregate measure, the dynamic query engine routes the query to the corresponding in-database aggregate.

• Measures with Regular Aggregate type of Standard Deviation, Median, Variance, or Unknown.
  Because these aggregate types are processed by the dynamic query engine, there are no corresponding aggregate recommendations for these types of measures.

• Measures with a Visible property set to False
  Aggregate Advisor recommendations focus on measures that are likely to be queried by users. Non-visible measures are not directly available to users.
For more information on using Aggregate Advisor, see the IBM Cognos Dynamic Query Analyzer User Guide.
Chapter 5. Getting started with Cognos Cube Designer

IBM Cognos Cube Designer is the modeling tool provided with IBM Cognos Dynamic Cubes. You use it to build dynamic cubes and publish them for use in the IBM Cognos studios.

To get started, you import metadata from a relational database. Using the metadata, you model dynamic cubes and save the cube definitions in a project. After you publish the cubes, they are listed as data sources in Content Manager and their related packages are available to report authors.

Introduction to Cognos Cube Designer

IBM Cognos Cube Designer is the application that you use to model dimensional metadata and dynamic cubes. The Data Source Explorer tree, the Project Explorer tree, the object editors, and the Properties pane are the main parts of the Cognos Cube Designer user interface.

Getting Started page

The Getting Started page is shown when you start Cognos Cube Designer. You can also display this page at any time by clicking Show the Getting Started Page from the Help menu.

You can perform the following tasks:

- Click Create New from Metadata to import metadata into a new project. For more information, see “Import metadata” on page 41.
- Click Create New Blank Project to create a project. For more information, see “Managing a project” on page 45.
- Click Open Existing to open a project. For more information, see “Managing a project” on page 45.

Data Source Explorer

The Data Source Explorer shows the metadata that is imported from relational data sources. You can view the columns, keys, and joins by expanding a table in the Data Source Explorer tree.

You can perform the following tasks:

- Right-click a table and select Explore Metadata to view a graphical representation of the metadata in the Relational Explorer Diagram tab. You can view the columns in a table, the primary key and foreign keys, and its joins to other tables.
- Right-click a table and select View Data to view sample data from the data source in the Tabular Data tab. Data is retrieved from the data source and shown in IBM Cognos Viewer.
- Right-click a fact table and select Generate, cube with basic dimensions or Generate, cube with dimensions using data sampling to create a dynamic cube.
Use one of these options to create a dynamic cube that is based on a fact table in the data source. The cube, including all required dimensional metadata, is added to the project in the Project Explorer. For more information about creating cubes, see “Model a dynamic cube” on page 61.

**Relational Explorer**

The Relational Explorer Diagram shows a graphical view of your data source metadata. Use the Relational Explorer Diagram to explore your metadata and view the relationships between objects.

**Tip:** When this tab is visible, you can drag tables from the Data Source Explorer tree to explore them.

**Project Explorer**

The Project Explorer shows all the dimensional metadata definitions and dynamic cube definitions included in a project. Use the Project Explorer tree to add objects to your dynamic cubes, access the object editors, and publish your cubes.

You can perform the following tasks:

- Model dimensions and hierarchies.
  For more information, see Chapter 6, “Dimensional metadata modeling,” on page 47.
- Model dynamic cubes
  For more information, see “Model a dynamic cube” on page 61.
- Right-click and select Validate to validate an entire project or an individual object.
  For more information about validation, see “Validate a project and individual objects” on page 45.
- Right-click a cube and select Publish to deploy the cube and, optionally, publish a package to be used by report authors.
  For more information about publishing, see “Deploying and publishing dynamic cubes” on page 69.

**Tip:** When you add a dynamic cube to a project, the data source on which it is based is added to the Data Sources folder in the Project Explorer tree. You can view the database catalog and schema that is referenced by the data source in the Properties tab.

**Functions tab**

From the Functions tab, you have access to the operators, summaries, constants, and functions that you use in expressions.

**Object editors**

There is an editor available for each object. When an editor tab is visible, you can also access other functionality that is related to the object. For example, when viewing the cube editor, you have access to the Aggregates, Security, and Implementation tabs.
To access an editor and its related tabs, right-click the object in the Project Explorer tree and select Open Editor.

**Tip:** To keep multiple editor tabs accessible, right-click the tab and select Pin. Because some of the editor windows are similar in appearance, verify your edit location on the tab.

**Implementation tab**

The Implementation tab shows a physical diagram of the current object. For example, to view the implementation of an entire cube, right-click the cube in the Project Explorer tree tab, select Open Editor, and then select the Implementation tab. For some objects, you can also add or edit relationships between the cube objects. Select an object and click to use menus to explore the diagram.

**Object properties**

On the Properties tab, you can view and edit the properties of an object.

To access the properties of an object, select the object in the Project Explorer tree. For more information about object properties, see Chapter 6, “Dimensional metadata modeling,” on page 47 and “Model a dynamic cube” on page 61.

**Validation issues**

The Issues tab shows modeling errors and warnings for objects that must be fixed to validate them.

The Performance Issues tab shows a list of all the performance issues for objects. These issues affect how well a dynamic cube performs when it is published and started.

You can view validation issues for all objects in a project or for an individual object. Select the project or object in the Project Explorer tree and then click the Issues tab. For more information about validating objects, see “Validate a project and individual objects” on page 45.

**Import metadata**

You import metadata to use as the basis for modeling dimensional metadata and dynamic cubes.

**Remember:** You must ensure that the data source from which you import data supports the dynamic query mode.

You can import metadata from the following sources:

- A Content Manager data source.
  - Select this option to import metadata from a relational data source that is defined in IBM Cognos Business Intelligence.
- A Cubing Services model.
  - Select this option to import cube metadata from an IBM InfoSphere Warehouse Cubing Services model. IBM Cognos Cube Designer creates a separate dynamic cube definition for each cube that is contained in the InfoSphere Warehouse Cubing Services cube model.
Tip: If you want to browse hierarchy members from a data source while you are modeling dynamic cubes, before you import the metadata, check whether your Administrator created a development or test data source connection that contains a subset of the metadata. Using a smaller volume of metadata can speed up the modeling process.

**Importing metadata from a Content Manager data source**

If you want to model dimensional metadata and dynamic cubes based on a relational database, you import the metadata from a Content Manager data source.

You import metadata from one schema at a time. You must perform a separate import for each schema you want to use.

A separate file is created for each data source from which you import metadata. These files are stored in the `c10_location` data directory.

A dynamic cube is modeled using a single data source only. A project can contain many dynamic cubes, and if you have imported multiple data sources, each dynamic cube can be derived from a separate data source.

**Before you begin**

Before importing metadata from a Content Manager data source, check the following prerequisites:

- The data source contains a star or snowflake schema.
- The data source connection to the database uses a JDBC driver. This is required by dynamic query mode.
- The data source is defined in IBM Cognos Business Intelligence. If it does not exist, you must first create it.

For more information, see the *IBM Cognos Business Intelligence Administration and Security Guide*.

**Procedure**

1. From the Start menu, click Programs, IBM Cognos 10, IBM Cognos Cube Designer.
   You can also start the Cognos Cube Designer from IBM Cognos Framework Manager. From the Tools menu, select Run Cube Designer.
2. From the toolbar, click Get Metadata.
3. Click Browse Content Manager Datasource.
4. Select the database schema from which to import data, and then click OK.
   The imported metadata is shown as a list of database tables in the Data Source Explorer tree.

   Tip: If your project contains more than one imported data source, each data source is shown in a separate panel.
   You can now model dimensional metadata and dynamic cubes.
5. When you finish working, click Save.
Import InfoSphere Warehouse Cubing Services cube metadata

You can import cube metadata from an IBM InfoSphere Warehouse Cubing Services model. IBM Cognos Cube Designer creates a project with a separate dynamic cube for each cube that is contained in the imported model.

Cognos Cube Designer retains the basic structure of imported cubes and dimensions when you import cube metadata, but there are some differences in underlying InfoSphere Warehouse Cubing Services models that can cause issues during import. The following table describes these issues and suggestions to work around them.

Table 11. Import issues and suggested solutions

<table>
<thead>
<tr>
<th>Issue</th>
<th>Workaround</th>
</tr>
</thead>
<tbody>
<tr>
<td>InfoSphere Warehouse Cubing Services models use name-based member unique names (MUNs) to identify members whereas IBM Cognos Dynamic Cubes uses key-based MUNs.</td>
<td>In Cognos Dynamic Cubes, create the MUN expressions by using the supported Cognos expression syntax.</td>
</tr>
<tr>
<td>In InfoSphere Warehouse Cubing Services models, it is possible to create a dimension with multiple hierarchies and reference one single hierarchy in a cube. Cognos Dynamic Cubes does not support hierarchy selection, so all hierarchies are included by each cube that references the dimension.</td>
<td>In Cognos Dynamic Cubes, make a copy of the dimension and delete the unwanted hierarchies. You can then reference the new dimension in a dynamic cube.</td>
</tr>
<tr>
<td>Cognos Dynamic Cubes does not support shared attributes. As a result, only the first level that references the attributes contains the attributes. The other levels remain empty.</td>
<td>Delete the empty levels and, where appropriate, create the required attributes by dragging columns to the required levels.</td>
</tr>
<tr>
<td>In InfoSphere Warehouse Cubing Services models, expressions are created by using SQL. Cognos Dynamic Cubes converts the attribute references, but not the expression, to the dynamic query mode.</td>
<td>In Cognos Dynamic Cubes, create the expressions by using the supported Cognos expression syntax.</td>
</tr>
<tr>
<td>In InfoSphere Warehouse Cubing Services models, if a dimension has an attribute defined that is not part of any level, and the attribute is used to join to a fact table, Cognos Dynamic Cubes incorrectly imports the attribute, adds it to the lowest level, and marks the attribute as hidden.</td>
<td>In Cognos Dynamic Cubes, delete the incorrect attributes manually.</td>
</tr>
<tr>
<td>In InfoSphere Warehouse Cubing Services models, if the showMembers property is defined for a hierarchy, this property is lost during import.</td>
<td>In Cognos Dynamic Cubes, manually set the <code>Show Extraneous Padding Members</code> property.</td>
</tr>
<tr>
<td>In Cognos Dynamic Cubes, the default member of a hierarchy is not migrated during import.</td>
<td>Manually set the default member property.</td>
</tr>
<tr>
<td>Cognos Dynamic Cubes imposes restrictions on certain special characters that are used for cube names and other object names. If an unsupported special character is encountered, an error is shown.</td>
<td>Rename any InfoSphere Warehouse Cubing Services models to remove any unsupported characters before you import them.</td>
</tr>
</tbody>
</table>
Table 11. Import issues and suggested solutions (continued)

<table>
<thead>
<tr>
<th>Issue</th>
<th>Workaround</th>
</tr>
</thead>
<tbody>
<tr>
<td>In InfoSphere Cubing Services models, it is possible to define attributes under a measure dimension that can be used in expressions. Cognos Dynamic Cubes does not support this feature. Attributes are imported as query items in a measure dimension but are flagged as invalid.</td>
<td>Delete the query items in the measure dimension after you import an InfoSphere Warehouse Cubing Services model.</td>
</tr>
<tr>
<td>If security defined for InfoSphere Cubing Services models, it is lost during import.</td>
<td>In Cognos Dynamic Cubes, set up the required security definitions.</td>
</tr>
</tbody>
</table>

A log file is also created during the import process that includes details of any objects that cannot be fully imported.

**Performing an InfoSphere Warehouse Cubing Services model import**

You import cube metadata from an IBM InfoSphere Warehouse Cubing Services model into a project.

**Before you begin**

Before you begin importing an InfoSphere Warehouse Cubing Services model, check that the following tasks are complete:

- Ensure that the model is exported from the Design Studio in InfoSphere Warehouse Cubing Services.
- Ensure that the data source that is associated with the InfoSphere Warehouse Cubing Services model has a JDBC data source connection defined. This data source connection is required by the dynamic query mode.
- Ensure that the associated data source is defined in IBM Cognos Business Intelligence.

**Procedure**

1. From the Start menu, click Programs > IBM Cognos 10 > IBM Cognos Cube Designer. You can also start the Cognos Cube Designer from IBM Cognos Framework Manager. From the Tools menu, select Run IBM Cognos Cube Designer.
2. From the File menu, click Import Cubing Services Model.
3. Select the model from which you want to import metadata, and then click OK.
4. Select the data source connection that is associated with the InfoSphere Warehouse Cubing Services cube model, and then click OK.
   Cognos Cube Designer creates a project that contains one or more cubes based on the imported metadata.
   If there any issues with the imported metadata, a log file is created and a confirmation message is shown.
5. Click OK to acknowledge the message. You can then investigate the issues in the log file. By default, the log file is stored in installation_location\c10\logs
   You can now continue working in the project.
6. Click Save to save the project.
Managing a project

Dynamic cube definitions are saved in a project. This section describes how to open, edit, and save an existing project.

Tip: It is good practice to save a project at regular intervals.

Procedure

1. From the toolbar, click **Open**.
2. Select the project file (.fmd).
3. Click **OK**.
4. Edit individual objects as required.
   
   For more information, see Chapter 6, “Dimensional metadata modeling,” on page 47 and “Model a dynamic cube” on page 61.

5. When you finish, click **Save**.

Validate a project and individual objects

IBM Cognos Cube Designer automatically validates individual objects as you design them. Modeling issues are identified in the **Project Explorer** with icons shown next to objects that are causing the issues:

- Errors are indicated by a white cross on a red circle.
- Warnings are indicated by a yellow triangle.
- Performance issues are indicated by a gauge.

The **Issues** tab shows a list of all the modeling issues that are related to a selected object. You can click an issue for more details. If a solution is provided, you can resolve the issue by selecting the solution and clicking **OK**. You can also click **Invoke Editor** to access the object editor. Modeling issues affect the validity of a dynamic cube and prevent you from deploying it.

The **Performance Issues** tab shows a list of all the performance issues that are related to a selected object. These issues affect how well a dynamic cube performs when it is published and started. They do not affect the validity of a dynamic cube.

You can validate an entire project or an individual object at any time. Validate frequently and resolve issues as they are reported. If you attempt to model a large cube without validating as you go, you can have a long list of issues to resolve.

You can validate each object as you create it by right-clicking it in the **Project Explorer** and selecting **Validate**.

You cannot deploy a dynamic cube that contains modeling errors. It is possible to deploy a valid cube when the project contains unrelated objects that are not valid.
Chapter 6. Dimensional metadata modeling

You use IBM Cognos Cube Designer to model dimensions, hierarchies and levels.

Model dimensions

With IBM Cognos Cube Designer, you can model commonly used dimensions at the project level and reference them in one or more dynamic cubes. You can also model dimensions within a specific cube.

The following table lists the properties that you can set when modeling a dimension.

*Table 12. Properties of a dimension*

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>The dimension name shown in the IBM Cognos studios. If the project supports multiple locales, there can be versions of the name in all supported languages. For more information about multiple locales, see “Multiple locales” on page 82.</td>
</tr>
<tr>
<td>Comment</td>
<td>A comment or description of the dimension. Comments are not visible in the IBM Cognos studios.</td>
</tr>
<tr>
<td>Default Hierarchy</td>
<td>The hierarchy to use when no hierarchy has been specified for a dimension used in an expression. Applies only when multiple hierarchies are defined for a dimension.</td>
</tr>
<tr>
<td>Multilingual Support</td>
<td>Disabled (default) - Specifies that members do not have multiple locale support. By Column - Specifies that members support multiple locals. For more information about multiple locales, see “Multiple locales” on page 82.</td>
</tr>
<tr>
<td>Dimension Type</td>
<td>Regular (default) - Identifies a regular dimension. Time - Identifies a time dimension. For more information about relative time dimensions, see “Defining a relative time dimension” on page 77.</td>
</tr>
</tbody>
</table>

Defining a dimension

In IBM Cognos Cube Designer, you can model commonly used dimensions at the project level and reference them in one or more dynamic cubes. You can also model dimensions within a specific cube.

When you add a dimension, it contains an initial set of objects that you need to complete the dimension. When you validate the dimension, you can use information from the Issues tab to help you complete the dimension definition.
Procedure
1. Select the location from which you want create the dimension:
   - To create a shared dimension at the project level, select Model from the Project Explorer tree.
   - To create a dimension that is automatically linked to a dynamic cube, select the cube from the Project Explorer tree.
   The dimension is also shared at the project level.

   Tip: Use folders and namespaces to organize objects. Using folders and namespaces makes it easier for you to locate objects and view the structure of a project in the Project Explorer.

2. Click New Dimension. The dimension contains a set of initial objects you can use to complete the dimension.

3. To create additional hierarchies, click New Hierarchy.

4. To create additional levels, click New Level.

5. On the Properties pane, set the default hierarchy.

6. To access the dimension editor, right-click a dimension from the Project Explorer tree and select Open Editor.

7. Change the order of the levels by clicking Move Up and Move Down.

What to do next
To complete the dimension, you must complete the definition of each hierarchy and level that belongs to the dimension. For more information, see “Defining a hierarchy” on page 51 and “Defining a level” on page 53.

   Tip: Right-click a relational table and select Explore Metadata. You can use the Relational Explorer Diagram to help you understand the structure of the metadata used to design the hierarchies and levels.

When you finish modeling a dimension, you can perform the following tasks:
- Browse members from the data source. For more information, see “Browsing members” on page 57.
- Add a shared dimension to a dynamic cube by dragging and dropping it onto the dynamic cube in the Project Explorer tree.
Defining a dimension based on a relational table

In IBM Cognos Cube Designer, you can generate commonly used dimensions at the project level and reference them in one or more dynamic cubes. You can also generate dimensions within a specific cube.

Generate, dimension using data sampling applies a heuristic algorithm that interprets relationships among the data to identify levels. Based on the data in the selected table, a hierarchy of levels is generated, based on the cardinality of the data and the column names.

If your data is clean and complete, the generated levels are more accurate. The algorithm does not detect multiple hierarchies.

Procedure

1. Select the location from which you want create the dimension:
   - To create a shared dimension at the project level, select **Model** from the **Project Explorer** tree.
   - To create a dimension that this automatically linked to a dynamic cube, select the cube from the **Project Explorer** tree.
     The dimension is also shared at the project level.
   
   **Tip:** Use folders and namespaces to organize objects. Using folders and namespaces makes it easier for you to locate objects and view the structure of a project in the **Project Explorer**.

2. Click **Generate, dimension with data sampling**.

What to do next

Review the generated dimension definition and, if required, manually modify it to reflect how you want to view your data.

**Tip:** Right-click a relational table and select **Explore Metadata**. You can use the **Relational Explorer Diagram** to help you understand the structure of the metadata that is used to design the hierarchies and levels.

When you finish modeling a dimension, you can perform the following tasks:

- **Browse members from the data source.** For more information, see “Browsing members” on page 57.
• Add a shared dimension to a dynamic cube by dragging and dropping it onto the dynamic cube in the Project Explorer tree.

Model hierarchies

IBM Cognos Dynamic Cubes supports level-based hierarchies and parent-child hierarchies. A single level-based hierarchy is automatically added when you create a dimension. You can also create multiple level-based hierarchies in a dimension.

For more information, see “Dimensions” on page 15 and “Hierarchies” on page 15.

Complete the hierarchy definition using the properties listed in the following table:

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>The hierarchy name shown in the IBM Cognos studios. If the project supports multiple locales, there can be versions of the name in all supported languages.</td>
</tr>
<tr>
<td>Comment</td>
<td>A comment or description of the hierarchy. Comments are not visible in the IBM Cognos studios.</td>
</tr>
<tr>
<td>Multiple root members</td>
<td>False (default) - the hierarchy uses a single root member at the top of the hierarchy. Selecting this option creates the All level at the top of the hierarchy. You can change the default caption of the top level by editing the Root Caption property.</td>
</tr>
<tr>
<td></td>
<td>True - the hierarchy contains multiple root members. Selecting this option deletes the All level that is automatically created at the top of the hierarchy.</td>
</tr>
<tr>
<td></td>
<td>If a hierarchy is single root, Cognos Cube Designer generates the root member. Because all members must belong to a level, the root member is in the All level.</td>
</tr>
<tr>
<td>Add Relative Time Members</td>
<td>False (default) - If the hierarchy belongs to a Time dimension, relative time members are not added to the hierarchy.</td>
</tr>
<tr>
<td></td>
<td>True - If the hierarchy belongs to a Time dimension, relative time members are added to the hierarchy.</td>
</tr>
<tr>
<td></td>
<td>For more information, see “Defining a relative time dimension” on page 77.</td>
</tr>
<tr>
<td>Default Member</td>
<td>The member value to use when evaluating member expressions, where no value is specified for a hierarchy.</td>
</tr>
<tr>
<td></td>
<td>If the default member is empty, the root member of the hierarchy is used.</td>
</tr>
<tr>
<td></td>
<td>To set a default member, drag the required member from the Members folder in the Project Explorer tree.</td>
</tr>
<tr>
<td>Root Caption</td>
<td>The caption of the root member at the top of the hierarchy shown in the IBM Cognos studios. If the project supports multiple locales, there can be versions of the caption in all supported languages.</td>
</tr>
<tr>
<td>Parent-Child</td>
<td>False - Indicates that the hierarchy does not use a parent-child structure.</td>
</tr>
<tr>
<td></td>
<td>This property cannot be edited.</td>
</tr>
</tbody>
</table>
Table 13. Properties of a hierarchy (continued)

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Show Extraneous Padding Members</td>
<td>False (default) - collapse multiple paths of padding members under a single member into a single path.</td>
</tr>
<tr>
<td></td>
<td>True - show multiple paths of padding members for a single member.</td>
</tr>
<tr>
<td></td>
<td>For more information, see “Extraneous padding members” on page 19.</td>
</tr>
<tr>
<td>Caption of Padding Members</td>
<td>The caption to use for padding members in the hierarchy.</td>
</tr>
<tr>
<td></td>
<td>Empty (default) - use a Null caption.</td>
</tr>
<tr>
<td></td>
<td>Parent’s caption - use the caption of the parent.</td>
</tr>
<tr>
<td></td>
<td>For more information, see “Padding members” on page 18.</td>
</tr>
</tbody>
</table>

Defining a hierarchy

In IBM Cognos Cube Designer, a single level-based hierarchy is automatically added when you create a dimension. You can also create multiple level-based hierarchies in a dimension.

Procedure

1. From the Project Explorer tree, select the dimension you want to work with.
   - To create a new hierarchy, click New Hierarchy.
   - To access the hierarchy editor, right-click a hierarchy that belongs to the dimension and select Open Editor.
2. Complete or modify the hierarchy definition using the Properties tab. Identify the Default Member and Root Caption if required.
3. Set the Show Extraneous Padding Members and Caption of Padding Members if required.
   For more information, see “Padding members” on page 18.
4. If an All level is not required, set the Multiple Root Members property to true.
5. To add levels to the hierarchy, drag levels from the Levels folder to the hierarchy.

Model levels

In IBM Cognos Cube Designer, each level in a dimension is defined by creating attributes, mapping those attributes to the relational database source and identifying which attributes are level keys.

When you create a hierarchy, an All level is created at the top of the hierarchy. An All level contains a single member that aggregates data from all members in the lower levels of the hierarchy. For example, an All level in a Region hierarchy aggregates data for all cities, in all states, in all regions.

Complete the level definition using the properties listed in the following table:
Table 14. Properties of a level

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>The level name shown in the IBM Cognos studios. If the project supports multiple locales, there can be versions of the name in all supported languages.</td>
</tr>
<tr>
<td>Comment</td>
<td>A comment or description of the level. Comments are not visible in the IBM Cognos studios.</td>
</tr>
<tr>
<td>Level Type</td>
<td>Identifies whether the level is regular or time-based.</td>
</tr>
<tr>
<td></td>
<td>Default: Regular</td>
</tr>
<tr>
<td>Current Period</td>
<td>An expression used to define the current period in a time-based level. The value of the expression is compared to the value of the level key attribute at the level.</td>
</tr>
</tbody>
</table>

Complete the definition of the level attributes using the properties listed in the following table:

Table 15. Properties of an attribute

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>The attribute name shown in the IBM Cognos studios. If the project supports multiple locales, there can be versions of the name in all supported languages.</td>
</tr>
<tr>
<td>Comment</td>
<td>A comment or description of the attribute. Comments are not visible in the IBM Cognos studios.</td>
</tr>
<tr>
<td>Expression</td>
<td>This property is available only for attributes created in the Cognos Cube Designer.</td>
</tr>
<tr>
<td>Column Name</td>
<td>The name of the associated column in the relational database.</td>
</tr>
<tr>
<td></td>
<td>If the Multilingual property is true, this value can be set. For more information, see &quot;Adding support for multiple locales to members and attributes&quot; on page 82.</td>
</tr>
<tr>
<td>Visible</td>
<td>Controls whether the object is visible in the published package.</td>
</tr>
<tr>
<td></td>
<td>Non-visible objects are typically used to represent intermediate values. These objects are not intended to be used for direct reporting. However a non-visible object is always present in the published package because the object might be needed by other objects in a dynamic cube.</td>
</tr>
<tr>
<td></td>
<td>Non-visible objects are not displayed in the metadata browser and are removed from the output of reports which contain references to them. For example, a report that references a non-visible object does not include output from that measure.</td>
</tr>
<tr>
<td></td>
<td>Default: True</td>
</tr>
<tr>
<td>Data Type</td>
<td>The data type of the associated column in the relational database.</td>
</tr>
<tr>
<td></td>
<td>This property cannot be edited.</td>
</tr>
<tr>
<td>Precision</td>
<td>The precision of the associated column in the relational database.</td>
</tr>
<tr>
<td></td>
<td>This property cannot be edited.</td>
</tr>
<tr>
<td>Scale</td>
<td>The scale of the associated column in the relational database</td>
</tr>
<tr>
<td></td>
<td>This property cannot be edited.</td>
</tr>
<tr>
<td>Property</td>
<td>Description</td>
</tr>
<tr>
<td>------------</td>
<td>------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Multilingual</td>
<td>This property appears only if support has been enabled for multiple locales in the dimension. For more information, see “Multiple locales” on page 82. False (default) - This attribute does not support multiple locales. True - This attribute supports multiple locales.</td>
</tr>
</tbody>
</table>

The **Level Unique Key** consists of one or more attributes whose values uniquely identify each instance of a level. For more information, see “Defining a level unique key” on page 54.

**Member Sort** consists of one or more attributes that provide information about the ordering of the members within a level. For more information, see “Defining the member sort order” on page 54.

### Defining a level

In IBM Cognos Cube Designer, you define levels to model the relationships in a hierarchy.

For each level, you assign or create attributes, map them to the relational data source, identify level keys and, optionally, define a sort order. You can also hide the attribute in the published package if required.

### Procedure

1. From the **Project Explorer** tree, select a dimension, and click **New level**.
2. To access the level editor, right-click the level in the **Project Explorer** tree, and select **Open Editor**.
3. To create an attribute, click **New Attribute**.

   **Tip:** To give the new attribute a more meaningful name, right-click it and select **Rename**.
4. To map a table column to the new attribute, select the required column from the **Data Source Explorer** tree and drop it onto the **Mapping** column.

   **Tip:** You can also create attributes by dropping table columns to the **Attribute** column.
5. Select the attributes assigned to **Member Caption** and, if required, **Member Description**. For more information about these special attributes, see “Attributes” on page 23.
6. You can define the **Level Unique Key** one of two ways:
   - If the level unique key is a single attribute, select the **Level Unique Key** check box for the attribute.
   - If the level unique key is a composite key, click **Level Key**. For more information, see “Defining a level unique key” on page 54.
7. If required, specify the member sort order. For more information, see “Defining the member sort order” on page 54.
8. To hide an attribute in the published package, change the **Visible** property to false.
9. To assign the level to a hierarchy, select the level and drop it onto the hierarchy in the Project Explorer tree.

   Tip: You can also assign levels by dropping them into the hierarchy editor.

10. Expand the hierarchy in the Project Explorer tree and, if necessary, modify the order of the levels as they appear under the hierarchy.

### Defining a level unique key

The Level Unique Key consists of one or more attributes whose values uniquely identify each instance of the level.

A level key is meant to uniquely identify each of the members within a level. The first level key shown in the Level Key window is the business key and is denoted with the business key icon 🌐. The business key is significant as it generates the members. If a level key does not uniquely identify members within a level, then attributes from the current level or parent levels must be used to uniquely identify the members within the level.

For example, a City level can use a unique ID as its level key attribute. City names are not unique, so you cannot use the city name attribute as a level unique key. You can include the set of Region name, State name, and City name attributes as a composite level unique key because the three attributes together uniquely define a city.

Level keys in SQL statements retrieve data values from the database and the corresponding columns are used as the basis for grouping, joining, and filtering. For optimal performance, use an attribute with an integer data type as the level key. Avoid character and text fields. There can be a performance difference between an integer level key and any other numeric type depending on the database system in use. For more information, see “Levels” on page 22.

If the level unique key is a single attribute, select the Level Unique Key check box for the attribute.

If there are multiple level key attributes, the first attribute must be the level key for the level. You may have to reorder the attributes to ensure the appropriate attribute is defined as the level key.

**Procedure**

1. To define a composite level unique key, right-click a level in the Project Explorer tree, and select Open Editor.

2. Click Level Key 🌐.

3. Select the attributes that together uniquely identify the level.

4. Change the order of the attributes by clicking Move Up ⬆️ and Move Down ⬇️. The first attribute shown in the Level Key window must be the level key for the level.

### Defining the member sort order

By default, hierarchy members are shown in the order in which they are loaded into a dynamic cube.
You can select one or more attributes that define the sort order of the members within a level. For example, a Month level might have Month ID as a key attribute, Month Name as the caption attribute, and Month Number as an ordering attribute. Month Number is specified as the ordering attribute because Month Number sorts the months by calendar order whereas Month Name sorts the months alphabetically.

**Procedure**

1. Right-click a level in the Project Explorer tree, and select Open Editor.
2. Click Member Sort.
3. Select the required attributes from the Attribute column and click Add to add them to the Sorting column.
   You can change the sort order by selecting an attribute and clicking Move Up and Move Down.
4. To change the direction of sorting for an attribute, click the Direction column and select the required option.
5. Click OK.

**Model parent-child hierarchies**

In IBM Cognos Cube Designer, you model a parent-child hierarchy when the dimension data is based on a recursive relationship and it is not level-based.

For more information, see “Parent-child hierarchies” on page 20.

To model a parent-child hierarchy, you create attributes, map them to the relational data source and identify which attributes represent the parent key and child key. The child key also acts as the member key.

The top level member in a parent-child hierarchy is determined as the member whose parent is Null.

You define a parent-child hierarchy within a parent-child dimension. Be aware of the following constraints:

- A dimension containing a parent-child hierarchy cannot include any other hierarchies.
- The attributes used for the parent key and member key cannot be composite keys.
- A parent-child hierarchy member cannot contain multiple parents.
  If the imported data source contains hierarchy members with multiple parents, you can use surrogate keys in the data source to overcome this issue.

To access the parent-child dimension properties, double-click a parent-child dimension from the Project Explorer tree.

Complete the parent-child dimension definition using the properties listed in the following table:
Table 16. Properties of a parent-child dimension

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>The dimension name shown in the IBM Cognos studios. If the project supports multiple locales, there can be versions of the name in all supported languages.</td>
</tr>
<tr>
<td>Comment</td>
<td>A comment or description of the dimension. Comments are not visible in the IBM Cognos studios.</td>
</tr>
<tr>
<td>Default Hierarchy</td>
<td>The parent-child hierarchy defined within the dimension.</td>
</tr>
<tr>
<td></td>
<td>This property cannot be edited.</td>
</tr>
<tr>
<td>Multilingual Support</td>
<td>Disabled (default) - Specifies that members do not have multiple locale support.</td>
</tr>
<tr>
<td></td>
<td>By Column - Specifies that members support multiple locales.</td>
</tr>
<tr>
<td></td>
<td>For more information about multiple locales, see “Multiple locales” on page 82.</td>
</tr>
</tbody>
</table>

To access the parent-child hierarchy properties, double-click a parent-child hierarchy from the Project Explorer tree.

Complete the parent-child hierarchy definition using the properties listed in the following table:

Table 17. Properties of a parent-child hierarchy

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>The parent-child hierarchy name shown in the IBM Cognos studios. If the project supports multiple locales, there can be versions of the name in all supported languages.</td>
</tr>
<tr>
<td>Comment</td>
<td>A comment or description of the parent-child hierarchy. Comments are not visible in the IBM Cognos studios.</td>
</tr>
<tr>
<td>Default Member</td>
<td>The member value to use when evaluating member expressions, where no value is specified for a hierarchy.</td>
</tr>
<tr>
<td></td>
<td>If the default member is empty, the root member of the hierarchy is used.</td>
</tr>
<tr>
<td></td>
<td>To set a default member, drag the required member from the Members folder in the Project Explorer tree.</td>
</tr>
<tr>
<td>Root Caption</td>
<td>The caption of the root member shown in the IBM Cognos studios. If the project supports multiple locales, there can be versions of the caption in all supported languages.</td>
</tr>
<tr>
<td>Parent-Child</td>
<td>True - Indicates that the hierarchy uses a parent-child structure.</td>
</tr>
<tr>
<td></td>
<td>This property cannot be edited.</td>
</tr>
<tr>
<td>Show Data Members</td>
<td>True (default) - show data members for non-leaf members in the hierarchy.</td>
</tr>
<tr>
<td></td>
<td>False - hide data members for non-leaf members in the hierarchy.</td>
</tr>
<tr>
<td></td>
<td>For more information, see “Data members” on page 20.</td>
</tr>
</tbody>
</table>
Table 17. Properties of a parent-child hierarchy (continued)

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Caption of Data Members</td>
<td>The caption to use for data members in the hierarchy.</td>
</tr>
<tr>
<td></td>
<td>Empty (default) - use a Null caption.</td>
</tr>
<tr>
<td></td>
<td>Parent's caption - use the caption of the parent.</td>
</tr>
</tbody>
</table>

To access properties of an attribute, select the attribute in the Attribute column in the parent-child hierarchy editor. For more information about attribute properties, see “Model levels” on page 51.

**Defining a parent-child hierarchy**

In IBM Cognos Cube Designer, you can model commonly used parent-child hierarchies at the project level and reference them in one or more dynamic cubes. You can also model parent-child hierarchies within a specific dynamic cube.

**Procedure**

1. Select the location from which you want create the parent-child hierarchy:
   - To create a shared parent-child hierarchy at the project level, select Model from the Project Explorer tree.
   - To create a parent-child hierarchy that is automatically linked to a dynamic cube, select the cube from the Project Explorer tree.
     The parent-child hierarchy is also shared at the project level.

2. Click New Parent-Child Dimension.
   A new parent-child dimension is created with a parent-child hierarchy.

3. Edit the dimension properties in the parent-child Properties pane.

4. Open the parent-child hierarchy editor.

5. From the Project Explorer tree, drag table columns to the Attribute column to create the hierarchy attributes.

6. Select the attributes assigned to the Parent key and Child key.
   These attributes are mandatory.

7. Select the attributes assigned to the Member Caption, and Member Description.
   The Member Caption attribute is mandatory.

8. If required, specify the member sort order. For more information, see “Defining the member sort order” on page 54.


10. If required, edit the properties of the attributes using the attribute editor Properties pane.

**Browsing members**

When you finish modeling a dimension containing a regular hierarchy or parent-child hierarchy, you can browse the dimension members from the data source.
Tip: A dimension must be valid before you can browse its members. If the dimension that you want to browse is contained in a dynamic cube, the cube must also be valid.

When viewing members in the Cognos Cube Designer, relative time members do not reflect the current period expressions defined in a project, but the members can be used in other expressions if desired. Current period expressions are used when the cube is started.

**Procedure**

1. From the **Project Explorer** tree, select the hierarchy for which you want to browse members.
2. Expand the **Members** folder.
   - The parent level dimension members are shown.
     - **Tip:** Depending on the volume of metadata included in the data source, it can be time consuming to browse the full list of members. You can cancel browsing by pressing the Escape key.
3. Expand a member to view its child members.
   - Repeat this step to view further child members.
4. If you make changes to a dimension or hierarchy, you must refresh the list of members to browse.
   - To refresh members for all hierarchies in a dimension, right-click the dimension and select **Refresh Members**.
   - To refresh members in a specific hierarchy, right-click the **Members** folder and select **Refresh**.

---

**Dimension filters**

In IBM Cognos Dynamic Cubes, you can create dimension filters to restrict the members available in a published dynamic cube when a dimension contains more data than is required by the cube.

For example, a Time dimension might contain data for the previous 10 years, but a dynamic cube might reference data for a single year only.

You can also use dimension filters to restrict data to only those members that contain a corresponding record in the fact table. For example, if a product has no sales figures because it is new, you can exclude it from the Product dimension. This example requires a filter expression such as `Fact.productId = Dim.employeeId`. You must also set the **Exclude facts without corresponding dimension keys** property to False.

If a dimension is large, filtering dimension data can also improve performance of a published dynamic cube.

**Important:** When you create a dimension filter, it is automatically applied to all dynamic cubes that reference the dimension. If you do not want to apply a dimension filter to a dynamic cube, you must duplicate the dimension, delete the dimension filter, and reference the duplicated dimension.

The following table lists the properties that you can set when you define a dimension filter.
Table 18. Properties of a dimension filter

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>The name of the dimension filter. Filters are not visible in the IBM Cognos studios.</td>
</tr>
<tr>
<td>Expression</td>
<td>Defines the value of the filter by using attributes or measures from the dimension.</td>
</tr>
</tbody>
</table>
| Exclude facts without corresponding dimension keys | Indicates whether to also filter fact data to ensure consistency in the summary data in a published dynamic cube. Default: True  
**Important:** Setting this option to True might reduce the performance.  
For example, you have dimension filter for the Time dimension to include data for year 2013 only. If the sales fact table also contains sales data for other years, and you do not set this property to True, users can see the sales data for all years in the summary data. |

**Dimension filters in aggregate cubes**

If your project contains an aggregate cube that references a dimension filter, there might be issues if the aggregate cube does not include the same attributes or measures that are specified in the filter expression. You must ensure that the data is valid for the aggregate cube.

**Defining a dimension filter**

In IBM Cognos Cube Designer, you define dimension filters within a dimension at the project level.

**Procedure**

1. From the Project Explorer tree, select the dimension for which you want to define a filter.
2. Select the Filters tab.
3. Click the New Filter icon.
4. Select the filter, and then complete the dimension filter properties.
Chapter 7. Dynamic cube modeling

With IBM Cognos Dynamic Cubes, you design and prepare dynamic cubes for use as data sources in the IBM Cognos studios.

The process to create dynamic cubes includes the following tasks:

- In IBM Cognos Administration, create a JDBC data source connection to your relational database.
  For more information, see the “Create a data source” topic in the IBM Cognos Business Intelligence Administration and Security Guide.
- In Cognos Cube Designer, import the metadata to use for modeling dynamic cubes.
- In Cognos Cube Designer, model the dimensional metadata.
- In Cognos Cube Designer, model the dynamic cubes.
- In Cognos Cube Designer, deploy individual dynamic cubes as OLAP data sources to Content Manager in IBM Cognos Business Intelligence.
- In Cognos Cube Designer, publish a package that includes a deployed cube.
  It is also possible to manually publish a package using IBM Cognos Framework Manager. You might publish manually, for example, if you want to create a package containing more than one dynamic cube. For more information about creating and publishing packages, see the IBM Cognos Framework Manager User Guide.
- In IBM Cognos Administration, configure the deployed cube for use as a data source by the Query Service.
- In IBM Cognos Administration, start the dynamic cube.

Model a dynamic cube

In IBM Cognos Cube Designer, you can define a dynamic cube manually or generate a dynamic cube based on a table in your relational database.

A basic dynamic cube contains the following items:

- A measure dimension that contains at least one measure
- At least one dimension
- At least one hierarchy and associated levels defined for each dimension
- Mappings between the measure and the dimensions
- Attributes that reference table columns either directly, by expressions, or by an expression that is a constant value

For more information, see “Dynamic cubes” on page 24.

When modeling a dynamic cube, the relationship between a measure and a dimension must be defined for each dimension in the cube. This relationship is defined by a measure-to-dimension join. For more information, see “Defining a measure-to-dimension join” on page 67.
### Table 19. Properties of a dynamic cube

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>The name of the dynamic cube. This is also used as the name of the data source that represents the cube. If the project supports multiple locales, there can be versions of the name in all supported languages. <strong>Tip:</strong> When creating a Framework Manager package for the dynamic cube, select this name from the list of data sources.</td>
</tr>
<tr>
<td>Comment</td>
<td>A comment or description of the dynamic cube. Comments are not visible in the IBM Cognos studios.</td>
</tr>
<tr>
<td>Remove non-existent tuples</td>
<td>True (default) - remove tuples from the cross join set that cannot contain any data.</td>
</tr>
<tr>
<td></td>
<td>False - do not remove tuples from the cross join set.</td>
</tr>
<tr>
<td></td>
<td>Applies when a dimension has multiple hierarchies and a report contains the cross join of two or more of those hierarchies.</td>
</tr>
<tr>
<td></td>
<td>With this feature enabled, only those tuples for which data may exist are retained from the cross join, improving report efficiency. A cross join of hierarchies from the same dimension may contain tuples for which no data can possibly exist. For example, in a Time dimension with two hierarchies, the cross join of [2011 Q1] and [2011 Aug] would be removed since [2011 Q1] and [2011 Aug] do not share a common month.</td>
</tr>
</tbody>
</table>

**Related tasks:**
- “Defining a dynamic cube manually” on page 63
- “Defining a dynamic cube based on a relational table”

**Defining a dynamic cube based on a relational table**

When you generate a dynamic cube, IBM Cognos Cube Designer creates a basic cube structure. The structure includes a measure dimension with measures, a set of dimensions, and appropriate mappings to the tables and columns in the database. To complete the dynamic cube definition, you resolve any issues and manually adjust the definition to meet your requirement.

**Before you begin**

By selecting a fact table, you can use one of two options to generate a dynamic cube.

- **Generate, cube with basic dimensions**
  
  This option generates one or more levels per dimension. Dimension tables are located by using the foreign primary key relationship and dimensions are created based on these dimension tables. If a single dimension table is detected, a single level is created by using the table columns as attributes of the level. If
more levels are required, create them manually and move the attributes to the new levels. If a snowflake dimension is detected, a level is created for each table in the snowflake. Measures in the measure dimension are generated by using numeric columns that are not foreign keys in the selected fact table.

- **Generate, cube with dimensions using data sampling**

  This option generates one or more levels per dimension. It applies a heuristic algorithm to interpret relationships among the data to identify levels. A hierarchy of levels is generated, based on the cardinality of the data and the column names. If your data is clean and complete, the generated levels are more accurate. The algorithm does not detect multiple hierarchies.

  **Tip:** If the table that you select has no relationship to other tables, Cognos Cube Designer provides the option to create a cube by using the selected table as a measure dimension, with any numeric columns added as measures.

Because Cognos Cube Designer needs foreign keys to determine relationships, only fact tables with foreign keys can be used to generate a dynamic cube. If your database does not use referential integrity, you can manually define a dynamic cube to meet your requirements. For more information, see "Defining a dynamic cube manually."

**Procedure**

1. Select a fact table in the Data Source Explorer.
2. Right-click and select a Generate option.
   - Generate, cube with basic dimensions.
   - Generate, cube with dimensions using data sampling.

**What to do next**

Review the generated cube definition and, if required, manually modify it to reflect how you want to view your data. Any object that causes a modeling issue or requires further design is identified in the Project Explorer and an icon appears next to the object. On the Issues tab, you might be presented with actions required to resolve these issues and validate the dynamic cube.

**Defining a dynamic cube manually**

Because IBM Cognos Cube Designer needs the information provided by foreign keys to determine relationships, only fact tables with foreign keys can be used to generate a dynamic cube. If your database does not use referential integrity, you can manually define a dynamic cube to meet your requirements.

Any object causing a modeling issue or requiring further design is identified in the Project Explorer and an icon appears next to the object. You can validate an entire project or an individual object at any time. An effective practice is to validate each object as you create it. Right-click an object in the Project Explorer tree and select Validate.

**Procedure**

1. Select a namespace in the Project Explorer tree.
2. Click New Cube.
What to do next

A measure dimension is automatically created. To complete your dynamic cube, define your measures, dimensions, hierarchies, levels and joins.

Model measures

Using IBM Cognos Cube Designer, you can define a measure manually or you can generate a measure based on a column in your relational database. A dynamic cube contains one measure dimension.

For more information, see "Measures" on page 27.

Table 20. Properties of a measure dimension

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>The measure dimension name shown in the IBM Cognos studios. If the project supports multiple locales, there can be versions of the name in all supported languages.</td>
</tr>
<tr>
<td>Comment</td>
<td>A comment or description of the measure dimension. Comments are not available to the studio users.</td>
</tr>
<tr>
<td>Default Measure</td>
<td>During report processing, if no measure is defined for evaluation of a value expression, the default measure is used. The default measure can be a regular or calculated measure. Default: the first measure added to the dynamic cube.</td>
</tr>
</tbody>
</table>

Table 21. Properties of a measure item

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>The measure name shown in the IBM Cognos studios. If the project supports multiple locales, there can be versions of the name in all supported languages.</td>
</tr>
<tr>
<td>Comment</td>
<td>A comment or description of the measure. Comments are not available to the studio users.</td>
</tr>
<tr>
<td>Expression</td>
<td>The expression can refer to measures in the dynamic cube. The expression cannot contain multidimensional dynamic query mode constructs. This property is only available for measure items that were created in Cognos Cube Designer.</td>
</tr>
<tr>
<td>Column Name</td>
<td>The name of the associated column in the relational database. This property cannot be edited.</td>
</tr>
</tbody>
</table>
Table 21. Properties of a measure item (continued)

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Visible</td>
<td>Controls whether the object is visible in the published package. Non-visible measures are typically used to represent intermediate values in the construction of a complex calculated measure. These measures are not intended to be used for direct reporting. However, a non-visible measure is always present in the published package because the measure might be needed by other objects in a dynamic cube. Non-visible measures do not display in the metadata browser and are removed from the output of reports which contain references to them. For example, a report that references a non-visible measure does not include output from that measure. The default measure cannot be hidden. Default: True</td>
</tr>
<tr>
<td>Data Type</td>
<td>The data type of the associated column in the relational database. This property cannot be edited.</td>
</tr>
<tr>
<td>Precision</td>
<td>The precision of the associated column in the relational database. This property cannot be edited.</td>
</tr>
<tr>
<td>Scale</td>
<td>The scale of the associated column in the relational database. This property cannot be edited.</td>
</tr>
<tr>
<td>Regular Aggregate</td>
<td>The primary method used to aggregate data for the measure. Default: Sum</td>
</tr>
<tr>
<td>Data Format</td>
<td>Set the default format properties for corresponding data type (number, currency, percentage) for the measure.</td>
</tr>
</tbody>
</table>

Defining a measure based on a relational column

In IBM Cognos Cube Designer, you can define a measure based on a relational column. To create measures, add a cube and then create measures in the measure dimension folder under the cube.

For information about creating calculated measures, see "Calculated members" on page 71.

Procedure

1. From the Project Explorer tree, expand your cube.
2. Right-click the measure dimension and select Open Editor.
3. From the Data Source Explorer, drop a table column onto the Editor pane.
   The mapping to the associated column is automatically created. The Property fields are initialized from the table column values.
Defining a measure manually

In IBM Cognos Cube Designer, you can define a measure manually by creating a mapping to a database column or to an expression. To create measures, add a cube and then create measures in the measure dimension folder under the cube.

For information about creating calculated measures, see “Calculated members” on page 71.

Procedure

1. From the Project Explorer tree, expand your cube.
2. Right-click the measure dimension and select Open Editor.
3. Click New Measure to add a blank measure.
4. To give the new measure a more meaningful name, right-click the new measure and select Rename.
5. You can complete the measure one of two ways:
   - To map the measure to a table column, drag a table column from the Data Source Explorer onto the Mapping field.
   - To map the measure to an expression, define an expression in Expression property on the Properties pane.

Defining aggregation rules

Each measure has a regular aggregation type. The Regular Aggregate property identifies the type of aggregation that is applied to the measure. Aggregation rules can be used in addition to the regular aggregate. They specify how semi-aggregate measures are aggregated with respect to information from the dimension.

When you import metadata, IBM Cognos Cube Designer assigns values to the Data Type, Precision, Scale and Regular Aggregate properties based on the relational object. For cube measures, you can define aggregation rules for each related dimension.

Aggregate rules are applied in this order:
1. The Regular Aggregate property is applied to dimensions that are included in the query but do not have assigned Aggregation Rules.
2. The Aggregation Rules are then applied to their specified dimensions, in the order that you specified the rules.
3. The report-level aggregation that is specified in the query.

For more information about measures and aggregation rules, see “Measures” on page 27.

Procedure

1. Select the Aggregation Rules tab.
2. Select a measure in the Measures pane.
3. Select a related dimension from the Dimension column.
4. Click Include to activate the aggregation rule for the dimension.
5. From the Aggregation Rule drop-down list, select the aggregation rule to be used for the selected dimension.
6. When you have finished adding aggregation rules for the dimension, use **Up**, **Down**, **Top** and **Bottom** to specify the order to apply the aggregation rules.

**Defining a measure-to-dimension join**

You can define a measure-to-dimension join in a dynamic cube when the level of a join does not match the level of the fact table. You must define the correct measure-to-dimension join to avoid double counting data from the fact table.

For example, a fact table can contain data at the Day level, but it can be joined to the Time hierarchy at the Week level. If the measure-to-dimension join is not defined, measure data equates to actual counts multiplied by the number of days in a week.

**Before you begin**

You must add the required dimension and measures to a dynamic cube before you can define a measure-to-dimension join. For more information, see "Model dimensions" on page 47 and "Model measures" on page 64.

**Procedure**

1. From the Project Explorer tree, right-click the cube and select **Open Editor**.
2. For each dimension, select **Edit**.
3. Specify the join by relating columns in the dimension to columns in the measure.
4. Specify the relationship operator.
5. If the join is at a higher grain than the lowest level of the dimension, clear the **Join is at the lowest level of detail for the dimension** check box.

**Note:** IBM Cognos Cube Designer cannot automatically detect that a join is at a higher grain than the lowest level of a dimension.

**Measure dimension filters**

In IBM Cognos Dynamic Cubes, you can create measure dimension filters to restrict the fact data available in a published dynamic cube when measures contains more data than is required by the cube.

If a dimension is large, filtering dimension data can also improve the performance of a published dynamic cube.

The following table lists the properties that you can set when you define a measure dimension filter.

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>The name of the measure dimension filter. Filters are not visible in the IBM Cognos studios.</td>
</tr>
<tr>
<td>Expression</td>
<td>Defines the value of the filter by using attributes and measures from the dynamic cube.</td>
</tr>
</tbody>
</table>
Measure dimension filters in aggregate cubes

If your project contains an aggregate cube that references a measure dimension filter, there might be issues if the aggregate cube does not include the same attributes or measures that are specified in the filter expression. You must ensure that the data is valid for the aggregate cube.

Defining a measure dimension filter

In IBM Cognos Cube Designer, you define measure dimension filters within a dynamic cube.

Procedure

1. From the Project Explorer tree, select the measure dimension for which you want to define a filter.
2. Select the Filters tab.
3. Click the New Filter icon.
4. Select the filter, and then complete the measure dimension filter properties.

Measure folders

In IBM Cognos Cube Designer, you can create measure folders within a measure dimension to organize measures and calculated measures. You can also create sub folders within a folder.

A measure folder does not have any value and cannot be included in expressions or calculations.

When you publish a dynamic cube, empty folders are not visible to report users in the IBM Cognos studios. A folder that contains only hidden measures or secured measures is treated as an empty folder.

Measure folders in virtual cubes

You can create measure folders in a virtual cube. If a source cube includes a measure folder, it is not included in the virtual cube, but any measures within the folder are added.

Creating a measure folder

You create measure folders at the cube level.

Procedure

1. From the Project Explorer tree, select a measure dimension, and click the New measure folder icon.
2. If required, create sub folders by selecting the measure folder, and clicking New measure folder.
3. Drag objects as required into the measure folders.
What to do next

You can change the sort order of objects in measure folders. For more information, see "Changing the sort order of measures and folders."

Sort measures and folders

In IBM Cognos Cube Designer, you can change the order in which measures, calculated measures, and folders are sorted within a measure dimension. You can also sort objects within a specific folder.

The default sort order for dynamic cubes is the order that is shown in the measure dimension. You can change the order by manually moving objects to any position that you require. You can also sort objects in ascending or descending alphanumeric order. Sorting applies within a single level of nesting only. If a folder contains sub folders, they are not included in an alphanumeric sort.

Important: The default sort order for dynamic cubes that are published with previous versions of Cognos Cube Designer is ascending alphanumeric order. If you update or republish the cube with version 10.2.1.1, it overrides the previous sort order with the new default order.

Cognos Cube Designer sorts measures according to the project design language, not the locale that is defined for measures and folders or the server content language.

Sorting in virtual cubes

You can sort measures, calculated measures, and measure folders in a virtual cube. If objects are sorted in a source cube, the sort order is not included in the virtual cube.

Changing the sort order of measures and folders

You sort measures, calculated measures, and measure folders at the cube level.

Procedure

1. To manually sort measure objects, drag them to the required position in a measure dimension in the Project Explorer tree.
2. To sort items alphanumerically, from the Project Explorer tree, right-click a measure dimension or folder in which to sort items, and click one of the following options:
   - Sort, Ascending
   - Sort, Descending

Deploying and publishing dynamic cubes

When you finish modeling a dynamic cube in IBM Cognos Cube Designer, you can deploy it as an OLAP data source to Content Manager. To work with a deployed cube in the IBM Cognos studios, you must also publish a Framework Manager package for it, configure the cube as a data source, and start the cube.

Important: You must validate a dynamic cube before you can deploy it.
You use the Publish option to deploy a dynamic cube. You can also perform the additional tasks required to publish a cube in one step.

- **Select all options**
  This option publishes a Framework Manager package for the deployed dynamic cube, then configures and starts the cube.

- **Publish the package in: My Folders**
  By default, the cube name is used as the Framework Manager package name. You can specify a different package name in the Package Name box.

  **Tip:** You can move the location of published packages by using IBM Cognos Administration.

- **Add the dynamic cube to the default dispatcher**
  This option configures the deployed dynamic cube as a data source.

- **Start the dynamic cube**
  This option starts the dynamic cube, if you also configure the cube as a data source.

- **Associate my account and signon with the cube datasource**
  This option allows you to use credentials to access the data source in the IBM Cognos Studios.
  Select if anonymous access is disabled. Your account must use associated credentials. Go to the Personal tab in the Set preferences dialog of the IBM Cognos Portal, and create your credentials.

**Important:** Because these options use default settings, they are intended for deploying and testing a dynamic cube in a development environment rather than a production environment.

**Procedure**
1. Open the project that contains the dynamic cube that you want to deploy and publish.
2. In the **Project Explorer** tree, right-click the required cube, and then select **Publish**.
3. Select the additional options required to publish the cube.
4. Click OK.

**Results**
When the deployment and publish process is complete, a confirmation message is shown.
Chapter 8. Advanced dynamic cube modeling

After you create a basic dynamic cube in IBM Cognos Cube Designer, there are numerous ways to enhance the functionality of the cube.

You can perform the following tasks:
• add calculated members and measures
• model relative time dimensions
• use multi-locales and associated formatting

Calculated members

Calculated members add business logic into dimensions by introducing members whose value is computed from the values present within the underlying data.

The new members are available for use without being added to the underlying relational data source. A calculated member is defined by a dimensional expression.

A calculated measure is a calculated member that belongs to the measure dimension. There are no behavior differences between calculated members and calculated measures.

For more information, see “Calculated members in reports” on page 137.

For information about relative time calculated members, see “Model relative time dimensions” on page 76.

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>The name that appears in the IBM Cognos studios. If the project supports multiple locales, there can be versions of the name in all supported languages.</td>
</tr>
<tr>
<td>Parent Member</td>
<td>Specifies the parent of the calculated member in the member tree.</td>
</tr>
<tr>
<td>Expression</td>
<td>Defines the value of the calculated member using other members and a valid set of multidimensional operators and functions.</td>
</tr>
</tbody>
</table>

Table 24. Properties of a calculated measure

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>The name that appears in the IBM Cognos studios. If the project supports multiple locales, there can be versions of the name in all supported languages.</td>
</tr>
<tr>
<td>Expression</td>
<td>Defines the value of the calculated measure using other members and a valid set of multidimensional operators and functions.</td>
</tr>
<tr>
<td>Data Format</td>
<td>Set the default data properties for each type of data.</td>
</tr>
</tbody>
</table>
Table 24. Properties of a calculated measure (continued)

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Visible</td>
<td>Controls whether the object is visible in the published package. Non-visible measures are typically used to represent intermediate values in the construction of a complex calculated measure. These measures are not intended to be used for direct reporting. However, a non-visible measure is always present in the published package because the measure might be needed by other objects in a dynamic cube. Non-visible measures are not displayed in the metadata browser and are removed from the output of reports which contain references to them. For example, a report that references a non-visible measure does not include output from that measure. Default: True</td>
</tr>
<tr>
<td>Regular Aggregate</td>
<td>The primary method used to aggregate data for the measure. Default: Sum</td>
</tr>
</tbody>
</table>

Author calculated member expressions

IBM Cognos Cube Designer validates the syntax of expressions. After a cube is started, the dynamic cube engine validates the semantics of the calculated member and calculated measure expressions. Any calculated member or expression which is not successfully validated during cube start is removed from the cube and is not available in the studios.

The expression editor does not limit functions to the valid ones for a specific context.

There are some restrictions that apply to IBM Cognos Dynamic Cubes calculated members.

Do not use the following relational constructs in expressions used to define calculated members:

- Value summary functions (Not Member Summary functions)
- Value Analytic functions (rank, first, last, percentile, percentage, quantile, quartile, distinct clause, prefilter clause) - (Summaries/Member Summaries)
- Value Summary functions (standard-deviation-pop, variance-pop, distinct clause, prefilter clause)
- All running- or moving- summary functions (Summaries)
- All FOR clauses in aggregate functions (Summaries/Member Summaries)
- Date/time constants (Constants)
- All business date/time functions (Business Date/Time functions)
- Like, lookup, string concat ‘||’, trim, coalesce, cast (Common Functions)
- MOD function (Common Functions)

Calculated member and measure examples

IBM Cognos Cube Designer allows for the definition of dimensional calculated members and measures. Such expressions were previously defined only in the
reporting environment. When defined in a dynamic cube, the calculated members
are accessible in all of the IBM Cognos Business Intelligence studios. You can use
calculated measures to determine constant or weighted values. You can create
calculated members that represent an N period rolling window of data relative to a
current period member.

**Constant and weighted allocation**

 Measures in base dynamic cubes must have the same grain since each base cube is
constructed from a single fact table. In a virtual cube, it is possible for a measure
from one base cube to be valid only for a subset of the levels of a virtual hierarchy.

In this example, the virtual cube Sales Inventory is built from two base cubes: Sales
and Inventory. The Sales cube has the measure Sales Amount and its Time
hierarchy contains Year and Quarter levels. The Time hierarchy in the Inventory
cube contains a Year, Quarter, and Month levels. When the Sales Inventory cube is
created, the virtual Time hierarchy contains the Year, Quarter, and Month levels.

In this situation, any Sales Amount value that is computed in the virtual cube at
the Day level is null since there is no value in the Sales cube at the Month level.

In the following diagram, the Sales Amount measure has no values at the Month
level but the Stock measure, from the Inventory cube, does. Only partial data is
used to show the hierarchy.

![Diagram of time hierarchy for two cubes](image)

**Figure 12. Example of differences in the time hierarchy for two cubes**

You can use calculated measures to compute constant or weighted values for a
measure such as Sales Amount. A constant allocation allocates a measure’s value
from a higher level evenly across all of its descendants at each level below the in
scope level. The in scope level is typically the lowest at which the measure is valid.

Using constant allocation, the following diagram shows the Sales Amount values.
The values from the Quarter level are evenly distributed across the descendants at
the Month level. Only partial data is used to show the allocation.

![Diagram of constant allocation](image)

**Figure 13. Example of the use of constant allocation**

A weighted allocation allocates values to the descendants relative to the values of
another measure that is in scope, and that is correlated with the measure being
allocated so the allocation is reasonable.
For example, the Sales Amount values are allocated based on the weights of the Stock measure from the Inventory cube.

Using weighted allocation, the following diagram shows the Sales Amount values. The values from the Quarter level are distributed using the same weighting as the Stock measure. Only partial data is used to show the allocation.

![Diagram showing weighted allocation]

**Figure 14. Example of the use of weighted allocation**

**Constant and weighted allocation expressions**

**Note:** To create the expression for a calculated measure, the database objects must be dragged from the Project Explorer into the Editor. In the example code, the bold text represents metadata objects such as hierarchies, levels, and measures that are dragged and dropped into the expression editor. The code is visible in the Expression property but cannot be entered as text.

The following expressions can be used to create calculated measures in the sample virtual cube gosldw_sales_and_target. Because the Sales Target data at the month level exists in the sample cube, these expressions are not necessary but are shown to illustrate how the expressions are constructed.

In this constant allocation example, the Sales Target measure is used.

```plaintext
if (roleValue('_levelNumber', currentmember([gosldw_sales_and_target].[Time].[Time])) > 2)
then
  (tuple([gosldw_sales_and_target].[Measures].[Sales target],
         ancestor(currentmember([gosldw_sales_and_target].[Time].[Time]),
                  [gosldw_sales_and_target].[Time].[Time].[Quarter]))
end
```

In this weighted allocation example, the Sales Target values are allocated based on the weights of the Revenue measure.

```plaintext
if (roleValue('_levelNumber', currentmember([gosldw_sales_and_target].[Time].[Time])) > 2)
then
  (tuple([gosldw_sales_and_target].[Measures].[Sales target],
         ancestor(currentmember([gosldw_sales_and_target].[Time].[Time]),
                  [gosldw_sales_and_target].[Time].[Time].[Quarter]))
else
  [gosldw_sales_and_target].[Measures].[Sales target]
```

74 IBM Cognos Dynamic Cubes Version 10.2.1.1: User Guide
N period rolling window

You can create calculated members that represent an N period rolling window of data relative to a current period member.

For example, to create a member that represents a six month rolling window of data, you can use the following calculated member expression. In the example code, the bold text represents metadata objects such as hierarchies, levels, and measures that are dragged and dropped into the expression editor. The expression aggregates the current measure over the set of members resulting from the set expression.

AGGREGATE (CURRENTMEASURE WITHIN SET LastPeriods(6, [Sales].[Time].[Time].[All Time].[Current Year (2013)]))

To create a member that represents a previous six month rolling window, you can use the following calculated member expression:

AGGREGATE (CURRENTMEASURE WITHIN SET LastPeriods(6, Lag([Sales].[Time].[Time].[All Time].[Current Year (2013)], 6))

Note: Either expression, especially the second, can return a set of less than six members depending upon how the time hierarchy is structured. Additional conditional logic is required to account for current time period members that are positioned less than the required number of periods (members) from the first member of the level in which they exist.

Relative time members contain a subset of the overall time hierarchy and are constrained in terms of which members are available for an N period rolling window. For example, you cannot represent a three year rolling window of data using the current year member since only the current year and prior year are available.

To access a larger window, you would require an expression such as:

AGGREGATE (CURRENTMEASURE WITHIN SET filter(MEMBERS([Sales].[Time].[Time].[Year]),
  _roleValue('_businessKey', currentmember([Sales].[Time].[Time])) =
  _roleValue('businessKey',
    [Sales].[Time].[Time].[All Time].[Current Year (2013)]))

This expression navigates from the sub tree of relative time calculated members, which are constrained to a set of current/prior periods, into the portion of the tree that contains the regular members of the hierarchy.
Defining a calculated member

You define calculated members in the expression editor using dimensional constructs and functions. You can define a calculated member based on a calculated member.

Calculated members are added to the member tree as a children of the parent member. You identify the parent member by selecting a member from the member tree under the Members folder of a hierarchy.

If there is no ALL member, the calculated member does not have to have a parent defined. The calculated member then becomes a member of the root level. If there is an ALL member, the calculated member must have a named parent and if one is not specified, the calculated member will fail to load. The failure is recorded in the log file.

It is a good practice to use a naming convention so that you and your report users can easily identify calculated members.

Procedure

1. From the Project Explorer, click a dimension and expand it.
2. Right-click a hierarchy that belongs to the dimension and select Open Editor.
3. Expand the hierarchy to access the Members folder.
4. Expand the member tree until you can view the member you wish to define as the parent of your new calculated member.
5. Select the Calculated Members tab.
6. Click New Calculated Member.
7. Select the new calculated member.
8. To set the Parent Member on the Properties pane, drag a member from the member tree in the Project Explorer. This property specifies the position of the calculated member in the member tree.
9. From the Properties pane, define the calculated member in the Expression property.
   - To use an object from the project, drag the item from the Project Explorer into the expression.
   - To use a calculated member, drag the calculated member from the member tree.
   - To add functions, summaries and operators, select the Functions tab to access the required elements.
10. Right-click the Members folder of the hierarchy and select Refresh.

Results

The new calculated member is displayed under the Calculated Members folder for the hierarchy. The calculated member can also be seen under the parent member in the Members folder of the hierarchy.

Model relative time dimensions

IBM Cognos Dynamic Cubes relative time members are specialized calculated members that are added to a time hierarchy at the time a cube is started.
IBM Cognos Cube Designer optionally creates a fixed set of relative time members in a time hierarchy. You can also create your own relative time calculated members.

**Defining a relative time dimension**

When you use relative time functionality, you can author reports relative to the current period. These reports can be executed at any time and remain valid based on the value of the current period at the execution time. To use relative time, you define a dimension as a time dimension, modify time properties for the level and generate relative time members on a hierarchy by hierarchy basis.

The predefined relative time members are Current Period, Prior Period, Current Period to Date, Prior Period to Date, Current Period to Date Change, Prior Period to Date Change, Current Period to Date Growth, and Prior Period to Date Growth. The predefined relative time members are used by simply dragging them into any report.

The modeler can create additional calculated members that are based on these members. The report author can create additional expressions that are based on these members.

Each level has a **Current Period** property. The current period property of a level is used to filter members by their level key value in order to identify the single leaf member which is the current period member in the hierarchy. This is the basis for defining the current member at each level in the hierarchy. If a Current Period expression is defined, it is used to filter members at that level by the value of the level key for that level. The current period value should map to the business key value of the member you want to be the current period member. The expression can be static, based on a current date/time value, or based on a value in the relational database typically populated by the ETL process.

The levels in a time hierarchy with relative time enabled are not required to have a Current Period. If no Current Period expressions are defined, the Current Period used is the rightmost, most recent leaf level member of the hierarchy.

The combination of level current period expressions is used to identify a specific leaf member. You can determine which member is used as Current Period by examining the levels of the hierarchy in a top-down manner. If there are levels with no current period expression defined, the member chosen at each level is the rightmost, most recent child of the member selected from the previous, higher level. As soon as a level is encountered where a Current Period expression is defined, the default selection of members at the higher levels is ignored and the member at that level which determines the path to the leaf level current period begins with the member defined by the expression. It is possible to define the current period of a hierarchy by providing a current period at the leaf level.

When viewing a time hierarchy, the caption of the relative time members that appear in the hierarchy do not use the current period expressions defined within the project. The product simply uses the right most/recent member at each level as the current period for that level.

Resolution of calculated member expressions is performed when a cube is started or member cache refreshed.
You cannot have a hierarchy with relative time members and non-default security rules.

**Procedure**

1. Select the location from which you want create the dimension:
   - To create a shared dimension at the project level, select **Model** from the **Project Explorer** tree.
   - To create a dimension that this automatically linked to a dynamic cube, select the cube from the **Project Explorer** tree.

   The dimension is also shared at the project level.

   **Tip:** Create one relative time dimension and use it in all your dimensions to avoid conflicts between multiple time dimensions.

2. Click **New Dimension**. The dimension contains a set of initial objects you need to complete the dimension.

3. On the **Properties** pane of the dimension, set **Dimension Type** to **Time**.

4. On the **Properties** pane of a hierarchy that belongs the dimension, set **Add relative time members** to **True**. This enables generation of the predefined relative time members.

5. Build your desired level structure. For more information about creating levels, see “Defining a level” on page 53.

6. For each time level, select a **Level Type**. The levels must appear in order in the hierarchy. For example, the Year, Month, Day levels cannot appear as Year, Day, Month. Use the **Periods** level type when the level does not conform to one of the predefined level types.

7. For each time level, enter an expression in the **Current Period** property.

   For some examples of current period expressions, see “Examples of level current period expressions.”

8. With the time dimension selected, right-click, and select **Refresh Members**. Predefined calculated members for relative time are added to the member tree.

IBM Cognos Cube Designer does not use of the **Current Period** expression when populating the relative time members. The most recent member at each level is used instead. However, the members may all be used within calculated member/measure expressions since the member identifiers remain constant; only their captions and what they refer to that changes.

### Examples of level current period expressions

Some common examples of level current period expressions are defined in the following list.

The expressions resolve to the value of the business key of the member that you want to be the current member.

#### Year

\[
\text{extract(year, localtimestamp)}
\]

#### \(\frac{1}{2}\) Year

\[
\text{if(extract(month, localtimestamp) < 7) then (1) else (2)}
\]
Quarter
'Q' || cast(
    if (extract(month, localtimestamp) <= 3) then (1)
    else ( if (extract(month, localtimestamp) <= 6) then (2)
    else ( if (extract(month, localtimestamp) <= 9) then (3)
    else (4) ) ) , varchar(1))

The current_timestamp function returns Greenwich Mean Time while the localtimestamp function returns local time.

Month
extract(month, localtimestamp)

Week of Year
cast(extract(year, localtimestamp), varchar(4)) || 'W' || cast(_week_of_year(localtimestamp), varchar(2))

Day of Year
cast(extract(year, localtimestamp), varchar(4)) || 'W' || cast(_week_of_year(localtimestamp), varchar(2))

Day of Week
_day_of_week(localtimestamp, 7)

Day of Month
_days_between(localtimestamp, _first_of_month(localtimestamp)) + 1

Hour
extract(hour, localtimestamp)

Week of Month
if( (_days_between( localtimestamp , _first_of_month(localtimestamp)) + 1) >
    _day_of_week(_first_of_month(localtimestamp), 7) )
then (1)
else (0)
+ 
if ((_days_between( localtimestamp , _first_of_month(localtimestamp)) + 1)
    _day_of_week(_first_of_month(localtimestamp), 7)) > 21)
then (4)
else if ((_days_between( localtimestamp , _first_of_month(localtimestamp))
    + 1) - _day_of_week(_first_of_month(localtimestamp), 7)) > 14)
then (3)
else (if ((_days_between( localtimestamp , _first_of_month(localtimestamp))
    + 1) - _day_of_week(_first_of_month(localtimestamp), 7)) > 7)
then (2)
else (1))

Calculated member example - creating a 24 month rolling window

The following example shows the process and code required for creating the expression to define a 24 month rolling window relative to the current time period. The final expression can be used within a calculated member created within the [Time] dimension in the IBM Cognos Cube Designer, making it available for use in all of the studios.

The expression ties a relative time member back to its corresponding member in the same hierarchy which is not a relative time member.

The reason for doing this is that relative time members are special calculated members which have parent/child and sibling relationships - but only relative to one another.
For example, you can obtain the children of the 'Current Year' member but you
cannot apply the PREVMEMBER function to the Current Year member to obtain
the prior year.

In the examples below, the expressions are formatted to make it easier to identify
the various functions and their operands.

In order to find the member which serves as the basis for the 'Current Year'
member, an expression such as following is required:

\[
\text{AGGREGATE (CURRENTMEASURE WITHIN SET ITEM(}
\text{FILTER(}
\text{MEMBERS{MyCube}.[Time].[Time].[Year],}
\text{ROLEVALUE('businessKey', CURRENTMEMBER{MyCube}.[Time].[Time]}
\text{) = ROLEVALUE('businessKey', [Current Year (2012)])}, 0, 0)))}
\]

The sample expression is based on the following conditions:
- the lowest level in the [Time] hierarchy is the [Month] level
- the time hierarchy contains the levels Year/Quarter/Month
- the Month key is based on the Year and Month level identifiers.

The expression to find corresponding member upon which [Current Month (Sept
2012)] is based, is as follows:

\[
\text{AGGREGATE (CURRENTMEASURE WITHIN SET ITEM(}
\text{FILTER(}
\text{DESCENDANTS(FILTER(}
\text{MEMBERS{MyCube}.[Time].[Time].[Year],}
\text{ROLEVALUE('businessKey', CURRENTMEMBER{MyCube}.[Time].[Time]}
\text{) = ROLEVALUE('businessKey', [Current Year (2012)])},
\text{[MyCube].[Time].[Time].[Month], SELF, ROLEVALUE('businessKey',}
\text{)}}
\text{))})
\]
The previous 24 months is then built upon that expression, making use of the LASTPERIODS function. This expression can be used within a calculated member created within the [Time] dimension in the Cognos Cube Designer.

\[
\text{AGGREGATE (CURRENTMEASURE WITHIN SET LASTPERIODS(24, ITEM(FILTER(DESCENDANTS(FILTER(MEMBERS([MyCube].[Time].[Time].[Year]), ROLEVALUE('businessKey', CURRENTMEMBER([MyCube].[Time].[Time]) = ROLEVALUE('businessKey', [Current Year (2012)])), [MyCube].[Time].[Time].[Month], SELF), ROLEVALUE('businessKey', CURRENTMEMBER([MyCube].[Time].[Time]) = ROLEVALUE('businessKey', [Current Year (2012)])), [MyCube].[Time].[Time].[Month], SELF), ROLEVALUE('businessKey', CURRENTMEMBER([MyCube].[Time].[Time]) = ROLEVALUE('businessKey', [Current Month (Sept 2012)])), 0, 0)))}
\]

This expression can be used within a calculated member created within the [Time] dimension in the Cognos Cube Designer, making it available for use in all of the studios.
Multiple locales

You can add support for multiple locales to IBM Cognos dynamic cubes. Metadata object names and captions, dynamic cube object names, and member attribute names can be assigned different values in different locales. Then when a user switches between different content languages in IBM Cognos Connection, names and captions are displayed in the appropriate language.

You use IBM Cognos Cube Designer to add support for multiple locales to a project, and you can then add metadata object names and member attribute names and captions in multiple languages. After adding multiple language support, you publish the dynamic cube in the normal fashion.

Selecting the design language and supported locales

When creating a project in IBM Cognos Cube Designer, the design language of the project defaults to the locale setting of the computer. You can change the default design language. Normally the default design language is the locale or language of the data in the database. After the design language is set, you can add other supported locales to the project.

Procedure

1. To change the design language, in the Properties tab of a project, click the value of the Design Language and select the design language from the drop-down list.
2. To add locales, in the Properties tab of a project, click Add Locale(s) and check the boxes next to the required locales.

Adding multiple locale names to metadata objects and dynamic cube objects

You can add names in multiple languages to metadata objects for supported locales.

Procedure

1. In Project Explorer, click a metadata object, such as a dimension, or a dynamic cube object, such as a measure.
2. In the Properties tab, click the value of the Name property. The supported locales for the project are displayed.
3. For each supported locale, enter a name for the object in that language.
4. You can add additional locales to the project by clicking the Add Language button. This adds locales to the project, not just to the selected object.
5. If the metadata object is a hierarchy, you can add language versions for the Root Caption property using the same steps.

Adding support for multiple locales to members and attributes

You add support for multiple locales for members and attributes by dimension. It is not necessary that all dimensions in a dynamic cube support multiple locales. IBM Cognos Dynamic Cubes supports dynamic cube definitions in which only some dimensions have members with multiple locales.
Before you begin

If you are adding multiple locales to attributes, the data source must contain a column for each locale associated with the attribute. For example, the Great Outdoors Warehouse data source has a Product line attribute in the Products dimension. This attribute has columns named PRODUCT_LINE_EN, PRODUCT_LINE_FR, and so on, for each of the supported locales in the database.

Procedure

1. In Project Explorer, click a dimension for which you want to add support for multiple locales.
2. In the Properties tab, click the value for Multilingual Support and select By Column.

   You can now provide multilingual names for members and attributes.
3. Perform the following steps for each member in the dimension that you want to give names in multiple languages.
   a. In Project Explorer, click a member in the dimension.
   b. In the Properties tab, click the value of the Name property. The supported locales for the project are displayed.
   c. For each supported locale, enter a name for the member in that language.
   d. You can add additional locales to the project by clicking the Add Language button.
4. Perform the following steps for each attribute in the dimension that you want to give names in multiple languages.
   a. In Project Explorer, click an attribute in the dimension.
   b. In the Properties tab, click the value of the Name property. The supported locales for the project are displayed.
   c. For each supported locale, enter a name for the attribute in that language.
   d. You can add additional locales to the project by clicking the Add Language button.
   e. In the Properties tab, change the value of the Multilingual property to true.
   f. In the Properties tab, click the value of the Column Name property. The supported locales for the project are displayed.
   g. Expand the data source in Data Source explorer, and drag the column associated with each locale into the respective Column Name value.

   For example, the Great Outdoors Warehouse data source has a Product line attribute in the Products dimension. This attribute has columns named PRODUCT_LINE_EN, PRODUCT_LINE_FR, and so on, for each of the supported locales in the database. If you are enabling multilingual support for a dynamic cube that uses the Product line attribute in this database, you would drag the PRODUCT_LINE_EN column into the Column Name value for English, the PRODUCT_LINE_FR column into the Column Name value for French, and so on.
Chapter 9. Aggregate cube modeling

In IBM Cognos Cube Designer, you can model aggregate cubes within a dynamic cube when the imported data source for a dynamic cube contains fact tables with pre-aggregated data.

For more information about pre-defined aggregate fact tables, see "Aggregate tables" on page 36.

After publishing a dynamic cube that contains aggregate cubes, when you run queries on the cube data source, IBM Cognos Dynamic Cubes analyses these queries and redirects them to the appropriate aggregate table in the data source.

You must be familiar with the fact data in the data source to model an aggregate cube. Understand which fact tables are set up as aggregates, and which detail tables the fact tables relate to.

**Tip:** It is good practice to prefix aggregate table names in the relational database with "Aggregate" so that you can easily identify them. You can also use the Relational Explorer to check the relationships between fact tables.

Before you can start modeling an aggregate cube, you must set up the dynamic cube and aggregate tables by performing the following tasks:

1. For level-based hierarchies only, create the hierarchy levels required for aggregation if they do not exist in the dimension. For example, if an aggregate table in the data source summarizes data by quarter, the Date dimension must include a Quarter level.
2. For each aggregation level in the dimension, ensure that the required attributes and level unique keys are defined.
3. Aggregate tables must contain data at the highest level of aggregation used by the aggregate cube so that you can roll up dimensions to the required level.

For example, if a Time dimension contains Year, Quarter and Month levels, and you want to roll up data up to the Year level in an aggregate cube, the aggregate table usually contains data at the Year level.

If Cognos Dynamic Cubes cannot match a rollup level to an aggregate table, it uses an aggregate table defined at a particular level of aggregation to satisfy higher level aggregate requirements. For example, if you want to roll up the Time dimension to the Year level, and the aggregate table only contains data at the Quarter level, it uses this aggregate table and rolls it up to the higher levels.

**Model aggregate cubes**

The way in which you model an aggregate cube depends on the data it contains:

- Simple aggregate table
- With a simple aggregate table, all fact data and levels keys are contained in a single table, so no joins to dimension data are required.

The aggregate table can be joined to the same dimension tables as the detailed fact table or joined to aggregate dimension tables. Aggregate tables do not contain the same level of detail as the non-aggregate dimension tables.
• Aggregate cube with a parent-child dimension
• A parent-child dimension does not have hierarchy levels. You create the relationships by mapping a single column in the aggregate table to the child key in the parent-child dimension.

You can partition data in an aggregate cube by using aggregate slicers. Partitioning is possible where the data source contains a set of aggregate tables, each providing a subset of the data set available. For example, an aggregate table can contain sales data for specific dates.

The following table lists the properties that you can set when you model an aggregate cube.

Table 25. Properties of an aggregate cube

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>The name of the aggregate cube. If the project supports multiple locales, there can be versions of the name in all supported languages.</td>
</tr>
<tr>
<td>Comment</td>
<td>A comment or description of the aggregate cube.</td>
</tr>
<tr>
<td>Remove non-existent tuples</td>
<td>This property is applicable to the dynamic cube only and must not be edited.</td>
</tr>
<tr>
<td>Ordinal</td>
<td>The order in which the dynamic query mode server redirects queries to an aggregate cube.</td>
</tr>
<tr>
<td></td>
<td>If there is only a single aggregate cube that can satisfy a query, the aggregate cube is used.</td>
</tr>
<tr>
<td></td>
<td>If there are multiple aggregate cubes that can satisfy a query, then the aggregate cube with the lowest cardinality at the lowest level of aggregation (ordinal value) is selected.</td>
</tr>
<tr>
<td></td>
<td>If there are multiple aggregate cubes with the same lowest ordinal value, then the aggregate cube that is defined higher in the list in IBM Cognos Cube Designer is selected.</td>
</tr>
</tbody>
</table>

For example, suppose you have the following aggregate cubes:
• aggregate cube 1, cardinality of 100, ordinal value 1
• aggregate cube 2, cardinality of 100, ordinal value 2
• aggregate cube 3, cardinality of 50, ordinal value 3
• aggregate cube 4, cardinality of 200, ordinal value 4
• aggregate cube 5, cardinality of 100, ordinal value 1

If a query can be satisfied by aggregate cubes 1, 2 or 3, then aggregate cube 3 is selected because it has the lowest cardinal value.

If a query can be satisfied by aggregate cubes 1, 2 or 4, then aggregate cube 1 is selected because it has a lower ordinal value than aggregate cube 2.

If a query can be satisfied by aggregate cubes 1 or 5, then aggregate cube 1 is selected because it is defined higher in the list in Cognos Cube Designer.
Defining an aggregate cube automatically

You can automatically define an aggregate cube when primary keys in the aggregate table match the level keys in dimensions of a dynamic cube. This allows you to create relationships between the dimensions and the aggregate table.

IBM Cognos Cube Designer can create these relationships automatically if the aggregate table contains the following:

- Measures that match the measures in the aggregate cube.
- Dimensions that match the dimensions in the aggregate cube.
- Data at the highest level of aggregation that is required by the aggregate cube.

**Procedure**

1. Open the Cube editor for the dynamic cube in which you want to define an aggregate cube.
2. Click the Aggregates tab.
3. Drag the required aggregate table from the Data Source Explorer to the Aggregates tab.

An aggregate cube is created in the Aggregates tab. The cube also appears under the Aggregate Cubes folder in the Project Explorer tree. Where matching measures and dimensions are found in the aggregate cube, Cognos Cube Designer maps each of these items to the aggregate table. Where possible, it also attempts to identify the highest level of aggregation that is required and roll up dimensions.

The ability to automatically map is dependent on how the aggregate tables are set up.

**Results**

The aggregate cube is now complete. You can fine-tune the mapping by following step 4 onwards in the topic "Defining an aggregate cube manually." When you finish, you can test the validity of the aggregate cube. For more information, see "Validate a project and individual objects" on page 45.

Defining an aggregate cube manually

You define an aggregate cube manually when an aggregate table uses level keys or is joined to a separate dimension that contains the required levels for aggregation. For example, to improve query performance, if a dimension table contains many records, you decide to create a dimension table that does not contain the lowest level members and contains only the level keys of its members. In this instance, you must map the relevant dimension in the aggregate cube to a separate dimension aggregate table.

**Procedure**

1. Select the dynamic cube in which you want to define an aggregate cube from the Project Explorer tree.
2. Click New Aggregate Cube.
3. Select the measures and dimensions to include in the aggregate cube, then click OK.

An aggregate cube is created, which also appears under the Aggregate Cubes folder in the Project Explorer tree.
By default, each dimension is mapped to the lowest dimension level defined in the detail fact table. If aggregation occurs at a higher level in the aggregate table, you must roll up dimensions in the aggregate cube to the correct level.

4. In the Project Explorer tree, double-click the aggregate cube in the Aggregate Cubes folder.
   The Aggregate cube editor is shown.

5. Click the dimension to roll up, and select the required level from the list of levels shown.
   Repeat this step for each dimension you want to roll up.
   For dimensions that are mapped to a separate dimension aggregate table, you must now map the level unique keys in the dimensions to columns in the required aggregate table.

6. In the Aggregate cube editor, click the Key Mappings tab.

7. For each level unique key, drag a column from the required aggregate table in the Data Source Explorer to the Mapping field.
   Tip: If you drag a whole aggregate table, IBM Cognos Cube Designer attempts to automatically map all level unique keys.
   Now you must map measures in the aggregate cube to columns in the aggregate table.

8. In the Aggregate cube editor, click Measures.
   The Measures editor is shown.

9. Map each measure to a column in the aggregate table by dragging a column from the required aggregate table in the Data Source Explorer onto the Mapping field.
   For those dimensions where primary keys in the aggregate table match the level unique keys in dimensions of the dynamic cube, you can now create the relationships between dimensions and measures in the aggregate cube.

10. In the Project Explorer tree, double-click the aggregate cube in the Aggregate Cubes folder.
    The Aggregate cube editor is shown.

11. For each dimension, click Edit, then select the dimension primary key and measure key to which it is joined.

12. If required, define the measure-to-dimension join in the Join is at the lowest level of detail for the dimension check box.
    For more information about this check box, see “Defining a measure-to-dimension join” on page 67.

13. Click OK.

Results

The aggregate cube is complete. You can now test the validity of the aggregate cube. For more information, see “Validate a project and individual objects” on page 45.

Defining an aggregate cube with a parent-child dimension

An aggregate cube can contain a parent-child dimension. Because the dimension does not have hierarchy levels, you create the relationships by mapping a single column in the aggregate table to the child key in the parent-child dimension.
The aggregate cube can also contain dimensions with level-based hierarchies. For more information about adding these dimensions, see "Defining an aggregate cube manually" on page 87.

**Procedure**

1. Select the dynamic cube in which you want to define an aggregate cube from the **Project Explorer** tree.
2. Click **New Aggregate Cube**.
3. Select the measures and parent-child dimension to include in the aggregate cube, then click **OK**.
   
   An aggregate cube is created, which also appears under the **Aggregate Cubes** folder in the **Project Explorer** tree.
   
   Now map a single column in the aggregate table to the child key in the parent-child dimension.
4. In the **Project Explorer** tree, double-click the aggregate cube in the **Aggregate Cubes** folder.
   
   The Aggregate cube editor is shown.
5. Select the parent-child dimension, then select the **I want to remap the columns for this dimension, as they are included in my aggregate** check box.
6. Click the **Key Mappings** tab.
7. Drag a column from the required aggregate table in the **Data Source Explorer** to the **Mapping** field for the child key.
   
   Next, you must map measures in the aggregate cube to columns in the aggregate table.
8. In the Aggregate cube editor, click **Measures**.
   
   The Measures editor is shown.
9. Map each measure to a column in the aggregate table by dragging a column from the required aggregate table in the **Data Source Explorer** onto the **Mapping** field.

**Results**

The aggregate cube is complete. You now test the validity of the aggregate cube. For more information, see "Validate a project and individual objects" on page 45.

---

**Filtering data using an aggregate slicer**

You can filter the data in an aggregate cube using aggregate slicers. Filtering is possible where the data source contains a set of aggregate tables, each one providing a subset of the data set available. For example, a data warehouse might contain five years of sales data, and also contain aggregate tables with sales data summarized for each quarter.

**Procedure**

1. Define the aggregate cube you require.
   
   For more information, see "Defining an aggregate cube automatically" on page 87, "Defining an aggregate cube manually" on page 87, and "Defining an aggregate cube with a parent-child dimension" on page 88.
2. Double-click the aggregate cube in the **Project Explorer** tree, then click the **Slicers** tab.
3. Select the data to include in the filter by dragging and dropping members from the Members folder in the Project Explorer tree to the Member Slicers field.

   Note: All selected members must come from a single hierarchy level.

Results

The aggregate cube is complete. You can now test the validity of the aggregate cube. For more information, see “Validate a project and individual objects” on page 45.
Chapter 10. Virtual cube modeling

Using IBM Cognos Cube Designer, you can model virtual cubes in a project.

For information about using virtual cubes, see "Virtual cubes" on page 32.

The following table lists the properties that you can set when modeling a virtual cube.

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>The name of the virtual cube. This is also used as the name of the data source that represents the cube. If the project supports multiple locales, there can be versions of the name in all supported languages. Tip: When creating a Framework Manager package for the virtual cube, select this name from the list of data sources.</td>
</tr>
<tr>
<td>Comment</td>
<td>A comment or description of the virtual cube. Comments are not visible in the IBM Cognos studios.</td>
</tr>
<tr>
<td>Merge Operator</td>
<td>The method used to aggregate data in the source cubes. Default: Sum. The cube merge operator is the default merge operator for all virtual measures and virtual members. You can also define a merge operator for a specific virtual measure or virtual member which overrides the cube merge operator.</td>
</tr>
</tbody>
</table>

Defining a virtual cube

You define a virtual cube at the project level.

**Procedure**

1. Select a namespace in the **Project Explorer** tree.
2. Click **New Virtual Cube**.
3. Select a maximum of two source cubes to merge into a virtual cube. You can include dynamic cubes from the current project, and dynamic cubes or virtual cubes deployed as data sources to the content store:
   - To include a dynamic cube from the project, select the cube from the list.
   - To include a dynamic cube or virtual cube from the content store, click **Add Content Store Cube**, select the required data source, then click **OK**.
4. Click **OK**.
5. Complete the virtual cube definition by using the **Properties** tab.
   You can view the source cubes from which the virtual cube is derived.
6. From the **Project Explorer** tree, right-click the virtual cube and select **Open Editor**. You can perform the following tasks from here:
   - To add a source cube, click **Add Source Cube**.
   - To delete a source cube, select the cube name, and click **Delete**.
To view the virtual measure dimension, click **Measures**.

### What to do next

You can now fine-tune virtual objects and define further objects as required. For more information, see "Model virtual dimensions," "Model virtual hierarchies" on page 93, "Viewing virtual levels" on page 95, "Model virtual members" on page 95, and "Model virtual measures" on page 97.

You can also add calculated measures or calculated members to a virtual cube. For more information, see "Calculated members" on page 71.

When you finish, you can test the validity of the virtual cube to check for errors, and then deploy and publish the virtual cube. For more information, see "Validate a project and individual objects" on page 45 and "Deploying and publishing dynamic cubes" on page 69.

**Tip:** If a virtual cube contains a source cube deployed as a data source to the content store, the data source must be started before you can deploy the virtual cube.

### Model virtual dimensions

When you create a virtual cube, IBM Cognos Cube Designer adds dimensions from the source cubes to the virtual cube.

Dimensions with identical names in the source cubes (conformed dimensions) are added to a virtual cube as merged virtual dimensions. Non-conformed dimensions are added to a virtual cube as new virtual dimensions. For examples of the merging process, see "Virtual cubes" on page 32.

If a virtual dimension is not merged correctly, or could not be automatically merged, you can manually merge two source dimensions. You can also delete redundant virtual dimensions.

When merging dimensions in a virtual cube, it is not possible to map a source dimension to more than one virtual dimension.

The following table lists the properties that you can set when modeling a virtual dimension.

#### Table 27. Properties of a virtual dimension

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>The name shown in the IBM Cognos studios. If the project supports multiple locales, there can be versions of the name in all supported languages.</td>
</tr>
<tr>
<td>Comment</td>
<td>A comment or description of the virtual dimension. Comments are not visible in the IBM Cognos studios.</td>
</tr>
<tr>
<td>Default Hierarchy</td>
<td>The hierarchy to use when no hierarchy been specified for a dimension used in an expression. Applies only when multiple hierarchies are defined for a dimension.</td>
</tr>
</tbody>
</table>
Table 27. Properties of a virtual dimension (continued)

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dimension Type</td>
<td>Regular (default) - Identifies a regular dimension.</td>
</tr>
<tr>
<td></td>
<td>Time - Identifies a time dimension. For more information about relative time</td>
</tr>
<tr>
<td></td>
<td>dimensions, see “Defining a relative time dimension” on page 72.</td>
</tr>
</tbody>
</table>

Defining a virtual dimension

Using IBM Cognos Cube Designer, you can define virtual dimensions within a virtual cube.

Procedure

1. From the Project Explorer tree, right-click the virtual cube and select Open Editor. The editor tab shows the following columns:
   - Virtual dimensions - the virtual dimensions added to the virtual cube.
   - Dimensions - the dimensions in the source cubes to which the virtual dimension is mapped.
2. To manually merge source dimensions into a new virtual dimension, follow these steps:
   a. Click Add Virtual Dimension.
   b. Click Editor for the source dimension column related to the new virtual dimension, then select a source dimension, and click OK.
   
   Tip: If you cannot select a source dimension because it is already mapped to a different virtual dimension, you must first delete the source dimension from the other virtual dimension.
   c. Repeat the step b for the second blank source dimension.
3. You can also perform the following tasks from here:
   - To delete a source dimension from a virtual dimension, select the source dimension, and click Delete.
   - To delete a virtual dimension from a virtual cube, select the virtual dimension, and click Delete.
4. To complete the definition of a virtual dimension, select the virtual dimension in the Project Explorer tree to display the Properties tab.

Model virtual hierarchies

When you create a virtual cube, IBM Cognos Cube Designer adds hierarchies from the source cubes to the virtual cube.

Hierarchies with identical names in the source cubes (conformed hierarchies) are added to a virtual cube as merged virtual hierarchies. Non-conformed hierarchies are added to a virtual cube as new virtual hierarchies. For examples of the merging process, see “Virtual cubes” on page 32.

If a virtual hierarchy is not merged correctly, or could not be automatically merged, you can manually merge two source hierarchies. You can also delete redundant virtual hierarchies.
When merging hierarchies in a virtual cube, it is not possible to map a source hierarchy to more than one virtual hierarchy.

The following table lists the properties that you can set when modeling a virtual hierarchy.

**Table 28. Properties of a virtual hierarchy**

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>The name shown in the IBM Cognos studios. If the project supports multiple locales, there can be versions of the name in all supported languages.</td>
</tr>
<tr>
<td>Comment</td>
<td>A comment or description of the virtual dimension. Comments are not visible in the IBM Cognos studios.</td>
</tr>
<tr>
<td>Default Member</td>
<td>The member value to use when evaluating member expressions, where no value is specified for a hierarchy.</td>
</tr>
<tr>
<td></td>
<td>If the default member is empty, the root member of the hierarchy is used.</td>
</tr>
<tr>
<td></td>
<td>To set a default member, drag the required member from the <strong>Members</strong> folder in the <strong>Project Explorer</strong> tree.</td>
</tr>
<tr>
<td>Parent-Child</td>
<td>True - Indicates that the hierarchy uses a parent-child structure.</td>
</tr>
<tr>
<td></td>
<td>False - Indicates that the hierarchy does not use a parent-child structure.</td>
</tr>
<tr>
<td></td>
<td>This property cannot be edited.</td>
</tr>
<tr>
<td>Add Relative Time Members</td>
<td>False (default) - The hierarchy does not belong to a Time dimension.</td>
</tr>
<tr>
<td></td>
<td>True - The hierarchy belongs to a Time dimension.</td>
</tr>
<tr>
<td></td>
<td><em>For more information, see <a href="#">“Defining a relative time dimension” on page 77.</a></em></td>
</tr>
</tbody>
</table>

**Defining a virtual hierarchy**

Using IBM Cognos Cube Designer, you can define virtual hierarchies within a virtual cube.

**Procedure**

1. From the **Project Explorer** tree, right-click the virtual dimension for which you want to define virtual hierarchies and select **Open Editor**. The editor tab shows the following columns:
   - Virtual hierarchies - the virtual hierarchies added to the virtual dimension.
   - Hierarchies - the source hierarchies in the source cubes to which the virtual hierarchy is mapped.

   **Tip:** If the virtual dimension was created from one source dimension only (not merged), only one source hierarchy column is shown.

2. To manually merge source hierarchies into a new virtual hierarchy, follow these steps:
   a. Click **Add Virtual Hierarchy**.
   b. Click **Editor** for the source hierarchy column related to the new virtual dimension, then select a source hierarchy, and click **OK**.
Tip: If you cannot select a source hierarchy because it is already mapped to a different virtual hierarchy, you must first delete the source hierarchy from the other virtual hierarchy.

c. Repeat the step b for the second blank source hierarchy.

3. You can also perform the following tasks from here:
   • To delete a source hierarchy from a virtual hierarchy, select the source hierarchy, and click Delete.
   • To delete a virtual hierarchy from a virtual cube, select the virtual hierarchy, and click Delete.

4. To complete the definition of a virtual hierarchy, select the virtual hierarchy in the Project Explorer tree to display the Properties tab.

Viewing virtual levels

When you create a virtual cube, IBM Cognos Cube Designer adds levels from the source cubes to the virtual cube.

Source cubes containing identical levels in a hierarchy are merged as virtual levels. If levels in the source cubes are not identical, level names from the first source cube are used as the names of the virtual levels. If one source cube contains more hierarchy levels than the second source cube, the extra levels are added as the lowest levels of the virtual hierarchy. For examples of the merging process, see “Virtual cubes” on page 32.

Procedure

From the Project Explorer tree, right-click the virtual hierarchy for which you want to view virtual levels and select Open Editor. The editor tab shows the following columns:
   • Virtual levels - the virtual levels added to the virtual hierarchy.
   • Levels - the source levels in the source cubes to which the virtual level is mapped.

Tip: If the virtual hierarchy was created from one source hierarchy only (not merged), only one source level column is shown.

Model virtual members

When you create a virtual cube, IBM Cognos Cube Designer adds members from the source cubes to the virtual cube.

For a virtual hierarchy that is merged from two conformed dimensions, all hierarchy members from the source cubes are available as virtual members. If the level key for each source member is identical, members are added to the virtual cube as merged virtual members. Any members that do not have matching level keys are added to the virtual cube as new virtual members. For examples of the merging process, see “Virtual cubes” on page 32.

If a virtual member is not merged correctly, or could not be automatically merged, you can manually merge two source members. You can also delete redundant virtual members.

When manually merging virtual members, if member names do not match, a new virtual member is created using this format: <source member 1?source member 2>. 
For example, two source cubes contain a Time hierarchy. Source cube 1 contains one member: All. Source cube 2 contains one member: All_Time. The virtual member created is All?All_Time.

**Tip:** Merged virtual names are required for internal member unique names (MUNs) only, and are not visible to report users.

The following table lists the properties that you can set when modeling a virtual member.

**Table 29. Properties of a virtual member**

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>The name that appears in the IBM Cognos studios. If the project supports multiple locales, there can be versions of the name in all supported languages.</td>
</tr>
<tr>
<td>Comment</td>
<td>A comment or description of the virtual member. Comments are not visible in the IBM Cognos studios.</td>
</tr>
<tr>
<td>Merge Operator</td>
<td>The method used to aggregate virtual members in the source cubes. By default, the merge operator is set to the same method that is defined for the virtual cube.</td>
</tr>
<tr>
<td>Precedence</td>
<td>The merge operator to use if a tuple contains virtual members with different merge operators. The merge operator with the highest precedence is used. If there are two or more merge operators with the same precedence, the merge operator for the first virtual member in the tuple is used. Default: 0</td>
</tr>
</tbody>
</table>

The following table lists the properties that you can set when working with a source member.

**Table 30. Properties of a source member**

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>The name that appears in the IBM Cognos studios. If the project supports multiple locales, there can be versions of the name in all supported languages.</td>
</tr>
<tr>
<td>Include</td>
<td>Controls whether the source member is included in the virtual cube. If the same member exists in two source cubes, and you exclude the member from both source cubes, the member is excluded from the virtual cube. If the member is excluded from one source cube only, the member is included in the virtual cube. Default: True</td>
</tr>
</tbody>
</table>

**Defining a virtual member**

Using IBM Cognos Cube Designer, you can model virtual members within a virtual cube.
Procedure

1. From the Project Explorer tree, right-click the virtual hierarchy for which you want to define virtual members and select Open Editor.

2. Select the Members tab. The editor tab shows the following columns:
   - Virtual members - the virtual members added to the virtual hierarchy.
   - Members - the source members in the source cubes to which the virtual level is mapped.

   Tip: If the virtual hierarchy was created from one source hierarchy only (not merged), only one source member column is shown.

3. To manually merge source members into a new virtual member, follow these steps:
   a. Click Add Virtual Member.
   b. Click Editor for the source member column related to the new virtual member, then select a source member, and click OK.

      Important: To see the list of source members in a hierarchy, the source cube must be deployed as data source to the content store and started.

   Tip: If you cannot select a source member because it is already mapped to a different virtual member, you must first delete the source member from the other virtual member.
   c. Repeat the step b for the second blank source dimension.

4. You can also perform the following tasks from here:
   - To delete a source member from a virtual member, select the source member, and click Delete.
   - To delete a virtual member from a virtual cube, select the virtual member, and click Delete.

5. To complete the definition of a virtual member, select the virtual member to display the Properties tab.

Model virtual measures

When you create a virtual cube, IBM Cognos Cube Designer adds measures from the source cubes to the virtual cube.

Measures with identical names in the source cubes are added to a virtual cube as merged virtual measures. Measures that do not have identical names or that exist in only one of the source cubes are added to a virtual cube as new virtual measures. For examples of the merging process, see “Virtual cubes” on page 32.

If a virtual measure is not merged correctly, or could not be automatically merged, you can manually merge two source measures. You can also delete redundant virtual measures.

When merging measures in a virtual cube, it is not possible to map a source measure to more than one virtual measure.

The following table lists the properties that you can set when modeling a virtual measure.
Table 31. Properties of a virtual measure

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>The name shown in the IBM Cognos studios. If the project supports multiple locales, there can be versions of the name in all supported languages.</td>
</tr>
<tr>
<td>Comment</td>
<td>A comment or description of the virtual dimension. Comments are not visible in the IBM Cognos studios.</td>
</tr>
<tr>
<td>Visible</td>
<td>Controls whether the measure is visible in the published package. Non-visible measures are typically used to represent intermediate values.</td>
</tr>
<tr>
<td></td>
<td>Non-visible measures are not displayed in the metadata browser and are removed from the output of reports which contain references to them.</td>
</tr>
<tr>
<td></td>
<td>For example, a report that references a non-visible measure does not include output from that object.</td>
</tr>
<tr>
<td></td>
<td>Default: True</td>
</tr>
<tr>
<td>Merge Operator</td>
<td>The method used to aggregate virtual measures in the source cubes. By default, the merge operator is set to the same method that is defined for the virtual cube, but you can override it.</td>
</tr>
<tr>
<td>Precedence</td>
<td>The merge operator to use if a tuple contains virtual measures with different merge operators.</td>
</tr>
<tr>
<td></td>
<td>The merge operator with the highest precedence is used. If there are two or more merge operators with the same precedence, the merge operator for the first virtual measure in the tuple is used.</td>
</tr>
<tr>
<td></td>
<td>Default: 0</td>
</tr>
<tr>
<td>Data Format</td>
<td>Set the default data properties for each type of data.</td>
</tr>
</tbody>
</table>

**Defining a virtual measure**

Using IBM Cognos Cube Designer, you can define virtual measures within a virtual cube.

**Procedure**

1. From the Project Explorer tree, right-click the virtual measure dimension and select Open Editor.
   
The editor tab shows the following columns:
   - Virtual measures - the virtual measures added to the virtual dimension.
   - Measures - the source measures in the source cubes to which the virtual measure is mapped.

2. To manually merge source measures into a new virtual measure, follow these steps:
   a. Click Add Virtual Measure.
   b. Click Editor for the source measure column related to the virtual measure, then select a source measure, and click OK.
Tip: If you cannot select a source measure because it is already mapped to a different virtual measure, you must first delete the source measure from the other virtual measure.

c. Repeat the step b for the second blank source measure.

3. You can also perform the following tasks from here:
   • To a source measure from a virtual measure, select the source measure, and click Delete.
   • To delete a source measure dimension (including all measures) from a virtual cube, select the source measure dimension, and click Delete.
   • To delete a virtual measure from a virtual cube, select the virtual measure, and click Delete.

4. To complete the definition of a virtual measure dimension or a virtual measure, select the required object in the Project Explorer tree to display the Properties tab.
Chapter 11. Define security

You can define security on hierarchy basis for a dynamic cube. Security is used to control the metadata that is available to specific users or user groups in the IBM Cognos studios. For example, if a dynamic cube includes a Geography hierarchy with two members, such as Canada and Europe, you can secure all members for Europe so that it is only accessible to certain users.

To define security, complete the following tasks as required:

- Define which members to secure in your hierarchies by creating one or more security filters for them.
  
  You can add security rules after modeling hierarchies in a project. They are independent of any dynamic cube.

- Apply security filters to a dynamic cube by creating one or more security views for them.

- Define which dimensions, attributes, and measures to secure in a dynamic cube by adding them to the security views.

- Publish a dynamic cube to the content store.

Tip: IBM Cognos Cube Designer validates security definitions when you validate or publish a dynamic cube.

After you publish a dynamic cube to the content store, you must complete the following tasks in IBM Cognos Administration:

- Assign users, groups, and roles to security views.
  
  This step is required if you are using role-based security filters.

  If you are using lookup filter-based security filters only, access rights are defined in the lookup tables, so it is only necessary to assign Read access to the user group named Everyone.

  For more information on role-based security filters and lookup table-based security filters, see "Security filters for hierarchy members."

- If you make further changes to security to a dynamic cube that is already started, refresh the security settings of the dynamic cube on the query service.

For more information on performing administration tasks, see Chapter 12, "Cognos dynamic cubes administration," on page 113.

Security for virtual cubes

Define security in source cubes as required. Virtual cubes automatically inherit the security settings that are defined in source cubes to maintain consistent security rules.

Security filters for hierarchy members

Secure members in a hierarchy by using a security filter. A security filter specifies whether you are granting or denying users access to one or more members.

You can add security rules after modeling hierarchies in a project. They are independent of any dynamic cube.
Every hierarchy in IBM Cognos Cube Designer contains a default security filter that is named **All Members Granted**. This option explicitly grants access to all hierarchy members. You can define further security filters as required.

There are two types of security filter you can create:
- **Lookup table-based security filter**
  If security rules for users are stored in a relational database table, you can import the data source and use the lookup table in a security filter.
- **Role-based security filter**
  You can also manually define security rules, for example, where no suitable lookup tables exist.

You can also combine lookup table-based and role-based security filters. For example, you can restrict access to sales data to the Sales Employees user group by using a security view, and then use IBM Cognos Administration to further restrict access for each salesperson in the lookup table.

For each filter, you must specify the scope to indicate whether you are explicitly granting or denying access to hierarchy members. You then complete the filter as follows:
- If you are defining a role-based security filter, you use a dynamic query mode expression to specify the required hierarchy members to include in the filter.
- If you are defining a lookup table-based filter, you specify which lookup table columns contain keys for hierarchy members from each level. You then use an expression to select the rows from the lookup table that are relevant to the user executing the query.

You can include macro expressions to match the user information in the lookup table to the current user information. An example is shown here:

```
(User Name = #sq($account.personalInfo.userName)#) and (Security Type = 'grant') and (Security Scope = 'self_and_descendant')
```

**Important:** A valid expression must return a set of hierarchy members.

In IBM Cognos Dynamic Cubes security, the deny scope has precedence over the grant scope. After a member is explicitly denied, it cannot be accessed. You can use a combination of deny filters to further restrict user access to the members in a hierarchy.

The following table describes the scope options that you can use when you are defining a security filter.

<table>
<thead>
<tr>
<th>Scope</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Grant Members</strong></td>
<td>Explicitly grant access to specified hierarchy members. Report users can see only the specified hierarchy members and associated values. Using a grant scope without the Ancestors option can lead to visible ancestors.</td>
</tr>
<tr>
<td><strong>Grant Members and Descendants</strong></td>
<td>Explicitly grant access to hierarchy members and all their descendants. Report users can see only the specified hierarchy members and associated values. Using a grant scope without the Ancestors option can lead to visible ancestors.</td>
</tr>
</tbody>
</table>
Table 32. Security filter scope options (continued)

<table>
<thead>
<tr>
<th>Scope</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Grant Members and Ancestors</td>
<td>Explicitly grant access to hierarchy members and all their ancestors. Report users can see only the specified hierarchy members and associated values.</td>
</tr>
<tr>
<td>Grant Members, Descendants and Ancestors</td>
<td>Explicitly grant access to hierarchy members together with all their descendants and ancestors. Report users can see only the specified hierarchy members and associated values.</td>
</tr>
<tr>
<td>Deny Members and Descendants</td>
<td>Explicitly deny access to hierarchy members and all their descendants. Report users cannot see the specified hierarchy members and associated values.</td>
</tr>
</tbody>
</table>

When you set up a security filter, you must consider the following points:

- When you explicitly grant access to a hierarchy member, report users can see only that member and its associated values. Users are denied access to all other hierarchy members.

  For example, the Geography hierarchy contains the following members: All, Canada, and Europe. If you grant access to the All member only, users cannot see Canada or Europe.

- When you explicitly grant access to a hierarchy member by using the Grant Members option or the Grant Members and Descendants option, report users can also see the ancestor members, but not their values.

  The values of these visible ancestor members are shown as ERR to differentiate them from a true Null value. The use of visible ancestors ensures that there is a path from a root member of the hierarchy to any granted members. Without a path from a root member to granted members, the IBM Cognos Studios cannot properly display members. Because Cognos Dynamic Cubes does not support visual totals, visible ancestors ensure that rollup values do not reveal information about secured descendants.

- When you explicitly deny access to a hierarchy member, access to all other members in the hierarchy, except descendant members, is implicitly granted.

- When you explicitly deny access to a hierarchy member, access to all descendant members is also denied.

  If the result of this option is an unbalanced or ragged hierarchy, padding members are used to balance the hierarchy. For more information, see “Padding members” on page 18.

- If a security filter is set up with a grant or deny scope option, but does not contain an expression, then no members are granted or denied.

- If a security filter contains references to a member that cannot be resolved, the member reference is ignored.

  If a member reference cannot be resolved because the member does not exist in a hierarchy, the security filter is still valid.

  If it cannot be resolved because the filter contains an invalid expression, an error occurs and access to the full hierarchy is denied.

- If an error occurs as a result of applying a security filter, when a user opens a package or runs a report, an error message is shown because access to the whole hierarchy is automatically denied.
Secured padding members

The use of secured padding members ensures that the hierarchies remain balanced. Balanced, non-ragged hierarchies have better performance in the studios. Secured padding members are inserted into a secured hierarchy member tree when a granted member has all its child members restricted. This scenario is most common with the Grant Members option, when descendants are not included in the scope. However, it can also occur with deny filters or with a combination of grant and deny filters.

Consider the following points:
- If all descendants of a non-leaf member are restricted, then secured padding members are inserted into all the levels below the non-leaf member.
- If all leaf members are restricted, padding members are inserted; the leaf level is not removed.
- The caption of secured padding members is either empty or blank or has the name of the parent. This is the same configuration setting for caption of padding member in ragged and unbalanced hierarchies.
- Secured padding members are secured similar to visible ancestors.
- Intrinsic properties of secured padding members are accurate, however member properties are null.
- There is at most one secured padding member for each level under a parent member.

Aggregated data in a secured dynamic cube

When you grant access to hierarchy members, it is possible that report users might inadvertently infer member values to which they are denied.

For example, suppose that you have a Geography hierarchy with these members and values: All (100), Canada (30), Europe (70). Using the Grant Members and Ancestors option, access is explicitly granted to Canada and its parent (All). Report users can see All (100), and Canada (30). If report users are aware that Europe is the only other hierarchy member, they can infer that its value is 70.

Default members

When a hierarchy is secured, a new default member on the hierarchy can be specified for the user. For example, if a single member and descendants is granted access, the default member can be modified. In this scenario, the single member is used as the new root of the hierarchy, although the member might not be at the root level.

The following steps determine the correct default member for a secured hierarchy:
- The original default member is checked to ensure that it is not restricted and not a visible ancestor. If the original default member is unsecured, then it remains the default member.
- A breadth first search of the hierarchy is done to find the first level with an unsecured member.
  - If the first level with an unsecured member has only the one unsecured member, then the unsecured member is the new default member.
  - If the first level with an unsecured member has more than one unsecured member, or also has a visible ancestor on the level, then their common
ancestor is the new default member. In some cases, this common ancestor can be a visible ancestor. In the case of a visible ancestor as a default member, any time a non-visible ancestor member is not the context in the report, the visible ancestor, whose value is always ERR, will be the context.

Any time a hierarchy with a visible ancestor as the default member is not explicitly included in the report, the default member is used in the context, and ERR is the cell values.

**Data caching using default members**

The same report run by a user with all access and a user with security policies will normally hit the same cache. In general, the secured user needs only a subset of the members that the unsecured user used, because security limits access to the members. However, when the default member differs between the two users, the slice of the cube differs, and a different section of the cache might be required.

The following example shows a cross-tab report of All Product against All Time on Quantity. The security views have the Branches hierarchy secured, but the Branches hierarchy is not included in the report. The default member for the Branches hierarchy is the slicer for the report.

In the case of the unsecured user, with a built-in scope of Grant All Members, the report uses the default member, All Branches, for the context of the Branches hierarchy. The tuple value searched for in the data cache is All Time, All Products, All Branches, Quantity.

| Table 33. Example of a cross-tab report using a default member of All Branches |
|---------------------------------|-----------------|
| Quantity                        | All Products    |
| All Time                        | 89,237,091      |

For the secured user that is assigned to a security view with a scope of Grant United States and descendants, the report uses the default member of United States, for the context of the Branches hierarchy. The tuple searched for in the data cache is (All Time, All Products, United States, Quantity). This differs from the unsecured user's tuple.

| Table 34. Example of a cross-tab report using a default member of United States |
|---------------------------------|-----------------|
| Quantity                        | All Products    |
| All Time                        | 10,444,575      |

Because the tuples are not the same, reports that are run by one user would not populate the tuple value in the data cache of the other. Also, because the Branches context is at different levels in the two tuples, the query structure to access the values in the underlying data source differs.

**Secure calculated members**

To secure calculated members, the members must be explicitly included in the dynamic query expression. Drag the calculated members into the expression editor to create a set expression that resolves to a set of members to be secured. For example, if you want to secure calculated members A1 and A2, drag them into the
editor and create an expression such as \( \text{SET}(A1, A2) \). Functions such as MEMBERS do not return calculated members that are present.

A calculated member is not accessible unless its parent member is accessible.

It might be possible that a calculated member definition references a secured member or measure. If a calculated member references a secured measure, a query with the calculated member returns the following exception: XQE-V5-0005 Identifier not found 'gosales_dw.Measures.Unit Sales'.

If the calculated member references a secured member, the value of the secured member is treated as null in the calculation.

**Security filters based on a lookup table**

If security rules for users are stored in a relational database lookup table, you can reference the lookup table in a security filter.

When you define a security filter, you specify the hierarchy levels on which to secure members. To specify the hierarchy levels, you map the level keys to one or more query items. You are not required to map all levels in a hierarchy. You map only those levels that you want to secure, and for which there is data in the lookup table. For each level with a multi-part level key, for example YearMonth for the Month level, you must map query items to all parts of the key.

The required combination of mapped query items depends on whether the level keys are unique or non-unique.

For example, suppose you have a Dates hierarchy with Year, Month, and Day levels, and you want to filter members at the Month level.

The following table illustrates that level keys are unique for each level.

*Table 35. Example hierarchy with unique level unique keys*

<table>
<thead>
<tr>
<th>Hierarchy Level</th>
<th>Level Key</th>
<th>Example Member Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Year</td>
<td>YearCode</td>
<td>2013</td>
</tr>
<tr>
<td>Month</td>
<td>MonthCode</td>
<td>201301</td>
</tr>
<tr>
<td>Day</td>
<td>DayCode</td>
<td>20130104</td>
</tr>
</tbody>
</table>

Because the level key uniquely identifies members at each level, you map only the level key for the Month level.

Consider the same hierarchy, but with non-unique level keys.

*Table 36. Example Dates hierarchy with non-unique level unique keys*

<table>
<thead>
<tr>
<th>Hierarchy Level</th>
<th>Level Key</th>
<th>Example Member Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Year</td>
<td>Year</td>
<td>2013</td>
</tr>
<tr>
<td>Month</td>
<td>YearMonth</td>
<td>January</td>
</tr>
<tr>
<td>Day</td>
<td>YearMonthDay</td>
<td>Friday</td>
</tr>
</tbody>
</table>

For each level, the unique level key is made up of the level key and parent level key. In this example, you must map the level keys for Year and Month.
You can define security for members at one or more levels of a hierarchy by using a single lookup table with null values. The lookup table must contain columns that correspond to the level keys for the levels that you want to secure.

For example, suppose a lookup table contains the columns Year, Quarter, and Month. The level keys are Year, YearQuarter, and YearQuarterMonth. If you reference the lookup table in a security filter, it can be used to identify members at any one of those levels. The following rows identify members from different levels:

- 2013, Null, Null identifies a year member.
- 2013, Q1, Null identifies a quarter member.
- 2013, Q1, Jan identifies a month member.

Each row in a lookup table corresponds to a one member at a single level. It should contain correct member key values in the columns corresponding to required level, and Null values in all other key columns. Incorrectly encoded keys are ignored.

**Tip:** The All member in a hierarchy does not have an associated level key value. To include the All member item, you must use Null values in all the lookup table key columns.

Before you can create the security filter, you must complete the following tasks:

- Import the metadata for the lookup table from the data source.
  
  For more information, see “Importing metadata from a Content Manager data source” on page 42.
- Model the lookup table by creating a query subject and adding query items to it.
  
  Each query item maps to a column in the lookup table.
  
  For more information, see “Modeling a lookup table.”

**Modeling a lookup table**
In IBM Cognos Cube Designer, you model a lookup table by creating a query subject at the project level.

**Procedure**
1. Select Model from the Project Explorer tree, and then click New Query Subject.
2. Right-click the query subject and select Open Editor.
3. Drag the required lookup table, or specific columns in the lookup table, from the Data Source Explorer onto the Editor pane.
   
   A query item is created for each column in the lookup table.

**Defining a security filter based on a lookup table**
After you have finished modeling a lookup table, you can define a security filter that is based on it.

**Procedure**
1. Select the hierarchy for which you want to define a security filter from the Project Explorer tree.
2. From the Security tab, click Add Lookup Table Based Security Filter.
3. Select the security filter, and then select the required option from the Scope list.
For information on scope options, see “Security filters for hierarchy members” on page 101.

4. Select the query subject that you defined for the lookup table from the Query Subject list.

5. Define the hierarchy level on which to filter by mapping the level keys to one or more query items in the Level Key Filters list.

6. Click Edit to define an expression to filter data in the lookup table.

   - For example, you can define an expression that restricts a report user to their own data only.

7. Define the filter expression. You can use the following methods to create the expression:
   - Select query items to include in the filter by dragging and dropping them from the query subject in the Project Explorer tree.
   - Type the expression manually, by using functions available from the Functions tab in the Project Explorer tree as required.

   For more information on using an expression in a security filter, see “Security filters for hierarchy members” on page 101.

   Tip: Query item references cannot be typed; they must be dragged and dropped.

8. Click Validate to check the expression is valid.

9. Click OK.

What to do next

To apply a security filter to a dynamic cube, you must now add the filter to a security view.

**Defining a role-based security filter**

You can manually define security rules for users where no lookup tables exist.

**Procedure**

1. Select the hierarchy for which you want to define a security filter from the Project Explorer tree.

2. Select the Security tab.

3. Click Add Role Based Security Filter.

4. Select the security filter, and then select the required option from the Scope list.

5. Click Edit to define an expression to add members to the security filter.

   - For example, you can define an expression that restricts a report user to their own data only.

6. Define the filter expression. You can use the following methods to create the expression:
   - Select members to include in the filter by dragging and dropping them from the Members folder in the Project Explorer tree.
   - Type the expression manually, by using functions available from the Functions tab in the Project Explorer tree as required.

7. Click Validate to check the expression is valid.

8. Click OK.
What to do next

To apply a security filter to a dynamic cube, you must now add the filter to a security view.

Security views

You apply security to a dynamic cube by defining a security view.

You can apply the following types of security to a view:

- hierarchy member security
  To apply hierarchy member security, you add one or more security filters to a security view.
  One view that contains a set of filters and a group of views that collectively contain the same set of filters should have the same view of a cube. The only difference is if tuples are not possible in an underlying view.

- measure, dimension, and attribute security
  To apply measure, dimension, and attribute security, you grant or deny access to the required objects in a dynamic cube.

There are several points to consider when you are setting up a security view:

- A security view that contains any explicit grant, including the built-in Grant All Members filter, takes precedence over a view that has no grant filters. A security rule might not have any grant filters if one of the following scenarios exists: if there are deny filters only for the hierarchy or if there are no filters that are defined for the hierarchy.

- If a security view contains a security filter that explicitly denies access to a hierarchy member, it is not possible for another security filter (in the same view or a separate view) to grant access to the same member.

- When you add multiple security filters to a security view, each filter is processed independently. If a security view does not include any security filters, users have access to all hierarchy members.

- If a security view contains multiple security filters, the resulting list of granted members is derived from merging all granted members minus all denied members.

- If there are no explicitly granted members, "all granted members" is replaced by all members in the hierarchy.

- Report users are granted access to an individual member only if that member is granted access in all individual security filters.

- When you merge security views by using IBM Cognos Administration, the resulting list of granted members is derived from merging all granted members minus all denied members.

- If there are no explicitly granted members, "all granted members" is replaced by all members in the hierarchy.

- Report users are granted access to an individual member only if that member is granted access in all individual security views.

- When a security view includes security filters that contain both grant and deny expressions, the resulting list of granted members is derived from merging all granted members minus all denied members.

- If report users are not assigned to any security view where security is defined, they are denied access to all hierarchy members.
Tuple security

IBM Cognos Dynamic Cubes dimensional security supports defining only which member users have access. There is no support for defining security on specific tuples or cells. However, if a user is in multiple views, it is possible that the combination of views will expose tuples that were not visible in any of the underlying views. If the tuple value is not visible in at least one of the underlying views, the tuple value will be ERR in the final view.

For a tuple value to be visible, the tuple must be visible in at least one of the underlying views.

Security view 1 contains granted United States, Outdoor Protection, and their descendants.

The table shows the tuple value.

Table 37. Example of a tuple value in a security view

<table>
<thead>
<tr>
<th>Quantity</th>
<th>Outdoor Protection</th>
</tr>
</thead>
<tbody>
<tr>
<td>Americas</td>
<td>United States</td>
</tr>
</tbody>
</table>

Security view 2 contains granted Brazil, Camping Equipment, and their descendants.

The table shows the tuple value.

Table 38. Example of a tuple value in a security view

<table>
<thead>
<tr>
<th>Quantity</th>
<th>Camping Equipment</th>
</tr>
</thead>
<tbody>
<tr>
<td>Americas</td>
<td>Brazil</td>
</tr>
</tbody>
</table>

Because the tuples (Brazil, Outdoor Protection) and (United States, Camping Equipment) are not visible in either of the underlying views, the tuples are indicated as errors in the final views.

The table shows the tuple value for the combined security views 1 and 2.

Table 39. Example of a tuple value for a combined security view

<table>
<thead>
<tr>
<th>Quantity</th>
<th>Camping Equipment</th>
<th>Outdoor Protection</th>
</tr>
</thead>
<tbody>
<tr>
<td>Americas</td>
<td>United States</td>
<td>---</td>
</tr>
<tr>
<td>Brazil</td>
<td>752,338</td>
<td>---</td>
</tr>
</tbody>
</table>

Defining a security view

You use IBM Cognos Cube Designer to define a security view for a dynamic cube.

Procedure

1. From the Project Explorer tree, right-click the required dynamic cube and select Open Editor.
2. Select the Security tab.
3. Click Add Security View.
What to do next

You can now add the required security filters and define which measures, dimensions, and attributes to secure.

Adding a security filter to a security view

You secure hierarchy members in a dynamic cube by adding the required security filters to a security view.

Procedure
1. From the Project Explorer tree, right-click the required dynamic cube and select Open Editor.
2. Select the Security tab.
3. Select the security view to which you want to add a security filter.
4. Select the Members tab.
5. Click Add Secured Member.
6. Select the security filters for each hierarchy that you want to secure, then click OK.

Defining secured measures

You secure measures in a dynamic cube by granting or denying access in a security view.

Procedure
1. From the Project Explorer tree, right-click the required dynamic cube and select Open Editor.
2. Select the Security tab.
3. Select the security view to which you want to add secured measures.
4. Select the Measures tab.
5. Click Add Secured Measures.
6. Select the measures for which you want to grant or deny access, and then click OK.
7. Select Grant or Deny as required for each measure that is listed in the Measures tab.

Defining secured dimensions

You secure dimensions in a dynamic cube by granting or denying access in a security view.

Procedure
1. From the Project Explorer tree, right-click the required dynamic cube and select Open Editor.
2. Select the Security tab.
3. Select the security view to which you want to add secured dimensions.
4. Select the Dimensions tab.
5. Click Add Secured Dimensions.
6. Select the dimensions for which you want to grant or deny access, and then click OK.
7. Select Grant or Deny as required for each dimension that is listed in the Dimensions tab.
Defining secured attributes
You secure attributes in a dynamic cube by granting or denying access in a security view.

Procedure
1. From the Project Explorer tree, right-click the required dynamic cube and select Open Editor.
2. Select the Security tab.
3. Select the security view to which you want to add secured attributes.
4. Select the Dimensions tab.
5. Click Add Secured Attributes.
6. Select the attributes for which you want to grant or deny access, and then click OK.
7. Select Grant or Deny as required for each attribute that is listed in the Dimensions tab.
Chapter 12. Cognos dynamic cubes administration

Dynamic cubes are published as OLAP data sources to IBM Cognos Content Manager. Administrators perform a number of tasks before dynamic cubes can be used by the IBM Cognos studios to create reports and analyses, and can perform additional tasks to manage or optimize the performance of dynamic cubes.

After the dynamic cubes are published as data sources, they can be accessed and configured in IBM Cognos Administration on the Status tab, in the Data Stores page. They can also be accessed from different areas in Cognos Administration; however, the Data Stores page is the central location where you can administer all instances of dynamic cube data sources in the IBM Cognos BI environment.

If you need information on publishing dynamic cubes, see “Deploying and publishing dynamic cubes” on page 69.

Administration tasks

Before you can work with published dynamic cube data sources, you must perform the following tasks:

- Assign an account in IBM Cognos to access the relational database that contains the data for the dynamic cubes.
- If you are using multiple dispatchers, define routing rules to ensure that reports are directed to the dynamic query server.
- Specify access permissions and capabilities required for modeling, configuring, managing, and optimizing dynamic cubes.
- Add dynamic cubes to the query service.
- Start dynamic cubes in the query service.

You can perform the following tasks to manage dynamic cubes or to optimize the performance of dynamic cubes:

- Assign users and groups to security views.
- Manage dynamic cubes. For example, you might refresh caches or security settings.
- Edit the query service configuration parameters for dynamic cubes. For example, you might need to edit the JVM (Java™ Virtual Machine) heap size.
- Edit dynamic cube properties. For example, you might change the default value for the data cache size limit.
- Create and schedule query service tasks.

After the dynamic cubes are used in reports and log files are analyzed, you can perform the following tasks:

- Use Aggregate Advisor to view aggregate recommendations.
- Monitor the metrics of the dynamic cubes added to the query service. For information about system performance metrics, see the IBM Cognos Business Intelligence Administration and Security Guide.
Access permissions and capabilities for dynamic cubes

Use the IBM Cognos groups, roles, and capabilities to define access permissions that are required for modeling, configuring, managing, and optimizing dynamic cubes.

To administer users, groups, and roles, the execute permissions are required for the Users, Groups, and Roles secured feature. The traverse permissions are required for the Administration secured function.

The following table describes the tasks that are associated with administering dynamic cubes:

<table>
<thead>
<tr>
<th>Task</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Model cubes</td>
<td>Model cubes in IBM Cognos Cube Designer and publish cube as a data source.</td>
</tr>
<tr>
<td>Secure cubes</td>
<td>Assign users and groups to security views in a dynamic cube data source.</td>
</tr>
<tr>
<td>Configure cubes</td>
<td>Assign a cube to a dispatcher and edit cube properties.</td>
</tr>
<tr>
<td></td>
<td>Assign a data access account.</td>
</tr>
<tr>
<td></td>
<td>Assign a server group to a dispatcher and packages, and define routing rules.</td>
</tr>
<tr>
<td>Optimize cubes</td>
<td>Run aggregate advisor and publish in-memory recommendations.</td>
</tr>
<tr>
<td>Manage cubes</td>
<td>Perform operations on cubes, such as starting, stopping, or refreshing data cache, and creating and scheduling query service tasks.</td>
</tr>
<tr>
<td>Prime cubes</td>
<td>Run scheduled jobs.</td>
</tr>
</tbody>
</table>

Each task requires a specific administration capability. The following table lists the administration capabilities and access permissions that are associated with specific tasks.

<table>
<thead>
<tr>
<th>Task</th>
<th>Administration capability</th>
<th>Access permissions required for the capability</th>
</tr>
</thead>
<tbody>
<tr>
<td>Model cubes</td>
<td>Data Source Connections</td>
<td>Read, Write, Traverse, Execute</td>
</tr>
<tr>
<td>Secure cubes</td>
<td>Data Source Connections</td>
<td>Read, Write, Traverse</td>
</tr>
<tr>
<td>Configure cubes</td>
<td>Configure and manage the system</td>
<td>Read, Write, Traverse</td>
</tr>
<tr>
<td>Optimize cubes</td>
<td>Data Source Connections</td>
<td>Read, Write, Traverse</td>
</tr>
<tr>
<td>Manage cubes</td>
<td>Administration Tasks</td>
<td>Read, Traverse, Execute</td>
</tr>
<tr>
<td>Manage cubes</td>
<td>Query Service Administration</td>
<td>Read, Write, Traverse, Execute</td>
</tr>
<tr>
<td>Manage cubes</td>
<td>Configure and manage the system</td>
<td>Read, Write, Traverse</td>
</tr>
<tr>
<td>Prime cubes</td>
<td>None</td>
<td>Read, Write, Traverse</td>
</tr>
</tbody>
</table>
Assigning the data access account for dynamic cubes

This account provides access to the relational database that the dynamic cube is based on.

The data access account is used to load data and metadata from the relational database, and to execute startup triggers in virtual cubes. The query service uses this account to log on to IBM Cognos Business Intelligence. The IBM Cognos account that you assign as the data access account must have access to the relational database that contains the source dynamic cube.

Before you begin

Before you assign the access account, perform the following tasks:

- Create trusted credentials for the user who will access the relational database that contains the source dynamic cube.
  
  For more information, see "Creating trusted credentials" on page 116.

- Create a data source signon for the user who will access the relational database that contains the source dynamic cube.
  
  The user ID and password that make up the signon must already be defined in the relational database.
  
  You can use multiple data source connections or multiple data source signons for dynamic cube data sources. In this situation, however, one of the connections and one of the signons must be defined using the name DynamicCubes.
  
  For more information, see "Creating a signon" on page 116.

For more information about creating data source connections and data source signons, see the IBM Cognos Business Intelligence Administration and Security Guide.

About this task

Virtual cubes do not require an access account because they obtain data from other source or virtual cubes. However, if a virtual cube has a startup trigger, it needs an access account. In this situation, the virtual cube uses the access account of the first source cube in the cube definition.

If a virtual cube is built by using two virtual cubes, it uses the access account that belongs to the first source cube of the first virtual cube.

Procedure

1. In IBM Cognos Administration, on the Status tab, click Data Stores.

   In the Scorecard section, you can see all published dynamic cube data sources.

2. For the dynamic cube for which you want to specify the access account, click the Actions drop-down menu, and then click Set properties.

3. On the General tab of the properties page, in the Access Account section, click Select the access account.

4. Browse the directory and select the user who will own the access account.

5. Click OK. The user name appears in the Access Account section.
Creating trusted credentials

You can create trusted credentials when you want to authorize other users to use your credentials because those users do not have sufficient access permissions to perform specific tasks.

For users to use trusted credentials, traverse permissions must be granted for the namespace.

Procedure

1. In IBM Cognos Connection, click the my area options button, My Preferences.
2. On the Personal tab, under Credentials, if you have not created credentials before, click Create the Credentials.
   
   Tip: If your trusted credentials are already created, you might only need to renew them by clicking Renew the credentials.
3. Select the users, groups, or roles that you want to authorize to use your credentials.
   If you are prompted for your credentials, provide your user ID and password.
4. If you want to add entries, click Add and choose how to select entries:
   • To choose from listed entries, click the appropriate namespace, and then select the check boxes next to the users, groups, or roles.
   • To search for entries, click Search and in the Search string box, type the phrase you want to search for. For search options, click Edit. Find and click the entry you want.
   • To type the name of entries you want to add, click Type and type the names of groups, roles, or users using the following format, where a semicolon (;) separates each entry:
     namespace/group_name;namespace/role_name;namespace/user_name;
     Here is an example:
     Cognos/Authors;LDAP/scarter;
5. If you want to remove an entry from the list, select the check box next to it and click Remove.

Results

The users, groups, or roles that can use your credentials are now listed in the Credentials section.

Creating a signon

The data source connection signon must be defined so that the query service can automatically access the data required for loading dynamic cubes.

About this task

A data source connection must have at least one signon that the query service can use to connect to the data source. If the data source connection has two or more signons, one of the signons must be named Dynamic Cubes. This signon will be used by the query service to connect to the data source.
**Procedure**

1. In IBM Cognos Administration, on the Configuration tab, click Data Source Connections.

2. Click the data source, and then click the connection to which you want to add a new signon.

3. Click the new signon button.

4. In the name and description page, type a unique name for the data source signon and, if you want, a description and screen tip, and then click Next.

5. Type the User ID and Password to connect to the database, and click Next. The Select the users page appears.

6. To add users and groups that can use the signon, and click Add.
   - To choose from listed entries, click the appropriate namespace, and then select the check boxes next to the users, groups, or roles.
   - To search for entries, click Search and in the Search string box, type the phrase you want to search for. For search options, click Edit. Find and click the entry you want.
   - To type the name of entries you want to add, click Type and type the names of groups, roles, or users using the following format, where a semicolon (;) separates each entry:
     
     namespace/group_name;namespace/role_name;namespace/user_name;

     Here is an example:
     
     Cognos/Authors;LDAP/scarter;

7. Click the right-arrow button and when the entries you want appear in the Selected entries box, click OK.

   **Tip:** To remove entries from the Selected entries list, select them and click Remove. To select all entries in a list, click the check box in the upper-left corner of the list. To make the user entries visible, click Show users in the list.

8. Click Finish.

   The new data source signon appears under the connection.

---

**Configure dynamic cubes for the query service**

The query service manages dynamic query requests and returns the results to the batch or report service that sent the request. You can configure one or more instances of the query service to run an instance of a dynamic cube.

You can perform most configuration and management actions for dynamic cubes from the Status tab, Data Stores page. On the Data Stores page, in the Scorecard section there are different views available: Data Stores - (All), Data Stores - Base Cubes, Data Stores - Virtual Cubes, and All server groups. To change the view, click the drop-down menu of the current view.

In the Data Stores - (All) view, you see a list of all dynamic cube data sources in the IBM Cognos BI environment, and in the Scorecard section you can see status information about cubes.

Cubes that are published to IBM Cognos Content Manager, but not configured show the status Unknown.
Cubes that are configured appear hyperlinked and show the status **Unavailable**. Note that cubes can appear hyperlinked, but Unknown for up to 30 seconds while the configuration process is being completed.

Cubes that are started show the status **Available**.

In the event that the query service is down, or communication between the dispatcher and the query service is unavailable, then the query service displays the status Unavailable and all cubes display the status Unknown.

Use the drop-down action menus for each data source to perform different actions on cubes. The actions that are available, depend on the status of cubes. The status and the action menus can be stale so you might need to click the **Refresh** icon to update the view.

You can drill down on each configured data source to the server groups for the cube and you can drill down again to the dispatchers. When you drill down to the dispatcher level, the **Metrics** section is populated with metrics for individual dynamic cubes. You can hover your cursor over each of the metrics to view descriptions of the metrics.

In the **All server groups** view you see a list of the query service groups to which cubes have been assigned. You can drill down on server groups to the dispatchers, and you can drill down again on the dispatchers for a list of all data sources served by a dispatcher. Use the drop-down action menus at each level to perform actions on cubes.

On occasion, when an action, such as changing query service properties for dynamic cubes, requires you to start or restart the query service, you must access the query service through the **System** page on the **Status** tab. The start and stop actions in the Data Stores tab are only used to perform actions on cubes.

**Using multiple dispatchers for the query service**

If you plan to use multiple dispatchers for the query service, you must define routing rules to ensure that reports are directed to the dynamic query server for execution. To ensure that your server processes dynamic cube requests, you need to complete the following tasks:

- Assign a server group to the dispatcher.

  **Tip:** To specify a server group name, on the **Status** tab in IBM Cognos Administration, click **System**. In the Scorecard section, choose the **All dispatchers** view. For each dispatcher, on its set properties page, click the **Settings** tab and choose **Tuning** under **Category**. For the Server group property, type a name of your choice in the **Value** box.

- Assign a routing set to all packages that are associated with a dynamic cube.

- Create a routing rule to send queries for the routing set to the server group.

Set routing rules in either IBM Cognos Administration or the IBM Cognos Software Development Kit. For more information, see the *IBM Cognos Business Intelligence Administration and Security Guide* or the *IBM Cognos Software Development Kit Developer Guide*. 
Adding dynamic cubes to the query service

Before you can start dynamic cubes, you must add them to the query service. You can add dynamic cubes to the query service individually or in groups.

Before you begin

You can add dynamic cubes to the query service by selecting the default server group.

If you are allocating dispatchers to dynamic cubes and are routing reports to select dispatchers within your Cognos BI environment, you need to create named server groups. For information on assigning dispatchers to server groups see "Configure dynamic cubes for the query service" on page 117.

If a set of virtual cubes and source cubes are part of the same hierarchy, you must add all cubes in the set to the same query service. For more information about hierarchies, see "Hierarchies" on page 15.

Procedure
1. In IBM Cognos Administration, on the Status tab, click Data Stores.
2. In the Scorecard section, select the Data Stores - (All) view.

   Tip: To change the view, click the drop-down menu for the current view.

3. Decide whether you want to add one or more dynamic cubes to a server group.
   - To add one dynamic cube, click its Actions drop-down menu and click Add data store to server group.
   - To add multiple dynamic cubes, select the check boxes for the applicable dynamic cubes. From the Group actions drop-down menu, click Add data store to server group.

4. In the window that is displayed, select the available server group or All.

   Tip: If the dynamic cubes that you are configuring are associated with dispatchers that share the same server group, add the cubes to this server group now. This helps to avoid problems with load balancing when you run reports based on these cubes.

5. View the results of your action in the response window. In the Scorecard section, the dynamic cube now appears hyperlinked.

6. In the Scorecard section, click the Refresh icon occasionally until the status of the cube changes to Unavailable. Configuration can take 30 seconds. When the cube is configured and has the status Unavailable, the drop-down action menu of the cube displays the Start action.

   Tip: The cube status and its action menu can be stale. To update the view, click the Refresh icon.

Results

When a dynamic cube is added to the query service, it is assigned the default configuration settings. You can change the default dynamic cube properties and query service properties. For more information, see "Setting dynamic cube properties" on page 125 and "Setting query service properties for dynamic cubes" on page 122.
After dynamic cubes are added to the query service, they must be started before they can be used by the IBM Cognos studios. For more information about starting cubes, see “Starting and managing dynamic cubes.”

If you need to remove dynamic cubes from the query service, use the Remove data store from server group action. The specified dynamic cube data sources will no longer be hyperlinked and the status will change to Unknown.

### Starting and managing dynamic cubes

The query service runs and creates an instance of a dynamic cube which is based on the model stored in Content Manager. Administrators can start, stop, refresh, and perform other actions to manage instances of dynamic cubes.

### Before you begin

Because virtual cubes are composed of source cubes there are several things to consider before you start, stop, and refresh the cubes:

- Virtual cubes and their source cubes must be available on the same dispatcher.
- Source cubes that are a part of a virtual cube must be started first.
- If source cubes are part of a virtual cube, the virtual cube must be stopped before the source cubes are stopped.
- When you refresh the data and member cache of a source cube, the data and member cache of any associated virtual cubes is also refreshed.
- You can only perform the following actions on virtual cubes: Start, Stop after active tasks complete, and View recent messages.

### About this task

You can perform most actions on individual or on multiple dynamic cubes. The actions that are available depend on the status of the cubes. Adding and removing cubes from server groups is described in the topic “Adding dynamic cubes to the query service” on page 119. The following list describes other actions that are associated with managing dynamic cubes in the query service.

**Start**  
This action starts dynamic cubes in the query service. You must start dynamic cubes in the query service to use them in IBM Cognos studios. When you start a cube, the hierarchy members are loaded in the cache.

Cubes started in the query service display the Available status in the Scorecard section of the Data Stores view. In some cases, when a cube is starting it will display the status Partially Available. The parent dynamic cube status reflects the consolidated status of child cubes.

**Set Properties**  
This action lets you set a number of general properties for dynamic cubes including hiding entries, and selecting the access account for the entry. For more information see “Setting general properties for a dynamic cube” on page 128.

**Stop after active tasks complete**  
This action stops cubes after the queries that are currently running are complete. Typically, you stop a cube if it does not need to be online and accessible.

**Stop immediately**  
Stopping immediately stops the cube and cancels any queries that are
currently running. This action is useful if you want to restart cubes to apply changes made to the model without waiting for long-running queries to complete.

**Restart**

This action stops and then starts a cube. For example, you might restart to reset a cube after a failure, or after a successful ETL (Extract, Transform, Load) run. Restarting a cube is a different action than restarting the query service. When following procedures, note whether the cube or the query service must be restarted.

**Refresh member cache**

If the dimension tables have been updated while the cube is running, you can refresh the member cache which will allow the cube to remain accessible to users while the dimension tables are reloaded from the back-end data source.

An update of the member cache, builds a new set of members in the background. This new set becomes available when the refresh is complete. This refresh requires additional memory to store two copies of the member cache in memory while the new cache is built.

Once the new member cache is available, the data cache is refreshed. This is because the data in the cache is tied to the structure of the members in the member cache.

For more information see "Types of caches used by dynamic cubes" on page 126.

**Refresh data cache**

Refreshing the data cache picks up changes to the fact table and re-syncs the data caches with the fact table. Data caches are refreshed dynamically while queries are still running so cubes remain accessible to users. When the member cache is brought online, a new corresponding data cache is also created. Even though a new data cache starts as empty, some additional space is required while the new cache is introduced and while queries are using the previous version of the data cache.

For more information see "Types of caches used by dynamic cubes" on page 126.

**Refresh security settings**

While the cube is still running, this action reloads the access permissions on the security views, and clears cached information that was loaded from the security lookup table.

This action also attempts to reload security rules from the model of a published cube. Rule reload succeeds only if the rest of the model was not significantly changed; for example, no levels, hierarchies or dimensions were added, changed, or removed. If these types of changes were done in the model, the rule reload is not executed and a corresponding message is written to the recent message log for the cube.

**Edit security view permissions**

Administrators can access the security view models for cubes, override the default group permissions, and add the appropriate users and groups to the model views. For more information see "Setting access permissions for security views" on page 131.

**Clear workload log**

This action removes all log entries for a dynamic cube. This is useful if you
want to capture new information about report usage. For more
information, see “Workload log for Aggregate Advisor” on page 127.

Delete This action deletes a published cube from Content Manager.

View recent messages
This action allows the administrator to view recent log messages to
diagnose problems with dynamic cubes. The time zone displayed is the
time zone of the administrator that is viewing the log messages.

Procedure
1. In IBM Cognos Administration, on the Status tab, click Data Stores.
2. In the Scorecard section, click the Data Stores - (All) view.
   • To perform an action on one dynamic cube, click the chosen action from the
cube Actions drop-down menu.
   • To perform an action on a group of dynamic cubes, select the check boxes
that are associated with the chosen cubes. Then from the Group actions
drop-down menu, select the action that you want to perform.
3. View the results of your action in the View the results window.

   Tip: The cube status and its action menu can be stale. To update the view, click
   the Refresh icon.

Results
For information about scheduling query service administration tasks, see “Creating
and scheduling query service administration tasks” on page 130.

Setting query service properties for dynamic cubes
The query service uses a number of environment, logging, and tuning
configuration settings.

About this task
When a dynamic cube is added to the query service, the default query service
configuration values are assigned to the cube. You can change the values to meet
the requirements of your IBM Cognos BI system.

Procedure
1. In IBM Cognos Administration, on the Status tab, select Data Stores.
2. In the Scorecard section, select the All servers groups view.

   Tip: To select a different view, in the Scorecard section, click the drop-down
   menu for the current view.
3. Click the server group under System.
4. From the Actions menu for the QueryService - dispatcher_name, click Set
   properties
5. Click the Settings tab.
6. In the Value column, type or select the values for the properties that you want
to change. The following list describes the properties that you can set for the
query service.
Advanced settings
Click Edit to specify advanced configuration settings. Because an entry acquires advanced settings from a parent, editing these settings overrides the acquired advanced settings. For information about types of advanced settings, see the IBM Cognos Business Intelligence Administration and Security Guide.

Dynamic cube configurations
Click Edit to add dynamic cubes to the query service.

Important: Starting with version 10.2.1 of IBM Cognos BI, the preferred way of adding dynamic cubes to the query service is documented in the topic “Adding dynamic cubes to the query service” on page 119.

Audit logging level for query service
Select the level of logging that you want to use for the query service.

Enable query execution trace
A query execution trace (run tree trace) shows queries that run against a data source. You use the trace to troubleshoot query-related issues.

You can find execution trace logs in the following location:
c10_location/logs/XQE/reportName/runTreeLog.xml

For more information about the query execution trace, see the IBM Cognos Dynamic Query Guide.

You can view and analyze these log files using IBM Cognos Dynamic Query Analyzer. For more information, see the IBM Cognos Dynamic Query Analyzer User Guide.

Enable query planning trace
Query plan tracing (plan tree) captures the transformation process of a query. You can use this information to gain an advanced understanding of the decisions and rules that are executed to produce an execution tree.

The query planning trace is logged for every query that runs using dynamic query mode. You can find planning trace logs in the following location: c10_location/logs/XQE/reportName/plantreeLog.xml

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

For more information about the query planning trace, see the IBM Cognos Dynamic Query Guide.

Generate comments in native SQL
Specifies which reports are generating the SQL queries in the database.

Write model to file
Specifies whether the query service will write the model to a file when a query runs. The file is used only for troubleshooting purposes. Modify this property only with the guidance of IBM Software Support.

You can find the file in the following location: c10_location\logs\model\packageName.txt

Idle connection timeout
Specifies the number of seconds to maintain an idle data source connection for reuse.

The default setting is 300. Valid entries are 0 to 65535.
Lower settings reduce the number of connections at the expense of performance. Higher settings might improve performance but raise the number of connections to the data source.

**Do not start dynamic cubes when service starts**
Prevents the dynamic cubes from starting when the query service starts.

**Dynamic cube administration command timeout**
Specify the amount of time to wait for a resource to be available for a dynamic cubes administration action. This action is canceled if the time period is exceeded.

**Tip:** Setting this value to zero causes the command to wait indefinitely.

**Minimum query execution time before a result set is considered for caching**
Specify the minimum amount of time to wait for a query before caching the results.

This setting does not apply to dynamic cubes.

**Initial JVM heap size for the query service**
Specifies the initial size, in MB, of the Java Virtual Machine (JVM) heap.

**JVM heap size limit for the query service**
Specifies the maximum size, in MB, of the JVM heap.

**Initial JVM nursery size**
Specifies the initial size, in MB, that the JVM allocates to new objects. The nursery size is automatically calculated. You do not need to change the setting unless IBM Cognos customer support recommends a change.

**JVM nursery size limit**
Specifies the maximum size, in MB, that the JVM allocates to new objects. The nursery size is automatically calculated. You do not need to change the setting unless IBM Cognos customer support recommends a change.

**JVM garbage collection policy**
Specifies the garbage collection policy used by the JVM. You do not need to change the setting unless IBM Cognos customer support recommends a change.

**Number of garbage collection cycles output to the verbose log**
Specifies the number of garbage collection cycles to be included in the verbose garbage collection. This controls the maximum size of the log file. Consult with IBM Cognos customer support to increase the setting and collect more logs.

**Disable JVM verbose garbage collection logging**
Controls JVM verbose garbage collection logging. You do not need to change the setting unless IBM Cognos customer support recommends a specialized change.

7. Start or restart the query service. For more information, see “Starting and stopping the query service” on page 125.

**Results**

A summary of the query service properties is displayed in the **Settings - Query Service** pane.
Starting and stopping the query service
When you change the query service configuration settings for dynamic cubes, you need to start or restart the query service for the changes to take effect.

Procedure
1. In IBM Cognos Administration, on the Status tab, select System.
2. In the Scorecard section, click the All Servers drop-down menu, point to Services and then click Query.
3. From the QueryService drop-down menu, click the required action.

Setting dynamic cube properties
Dynamic cubes are assigned default property values when they are added to the query service, but these values can be changed.

About this task
The default values are often the best choice, with the exception of the data cache size limit.

For more information on some of the properties, see "Types of caches used by dynamic cubes" on page 126 and "Workload log for Aggregate Advisor" on page 127.

Procedure
1. In IBM Cognos Administration, on the Status tab, click Data Stores.
2. In the Scorecard section, select the Data Stores - (All) view.
3. Click the dynamic cube that you want to modify, and then click the server group under the cube name.
4. For the QueryService dispatcher name, click the actions drop-down menu and click Set properties.
5. Change values for any of the following properties as required:
   - Disabled
     Disables the cube. This means that the cube is configured for a server, but is not running on that server.
   - Startup trigger name
     Type the name of the trigger event to send after this cube starts.
     When a cube is available for query processing, the event is triggered for execution against the server which triggered the event. The purpose of the event is to run reports to populate the cube cache with data.
   - Disable result set cache
     Disabling the cache is useful during the development or testing phase of a cube because it allows you to test the performance of the data cache.
   - Data cache size limit
     Type the maximum size of the data cache for the cubes.
     The default value is 1024 MB. The result of each query is written to disk. If the maximum size is exceeded, the older report sets are removed from the cache.
Maximum amount of disk space to use for result set cache
Type the maximum size of disk space.

The result of each query is written to disk. If the maximum amount of disk space is exceeded, the older report sets are removed from the cache.

Enable workload logging
Workload logging is used to capture information about queries that are sent to the dynamic query engine processes. This workload information is used by Aggregate Advisor to determine aggregate recommendations.

Maximum amount of memory to use for aggregate cache
Type the maximum size of the memory to use for the in-memory aggregates. In-memory aggregates are loaded when cubes are started and restarted, and when data cache is refreshed. The size of the aggregate cache is used in determining the total JVM heap size of the query service.

In-memory aggregates are loaded on a first come, first served basis. This means that if the aggregate cache is full, then no more in-memory aggregates can be loaded. Additionally, an in-memory aggregate may fail to load if the in-memory aggregate cache size limit would be exceeded if it did load.

The default value is 0, which specifies not to use in-memory aggregates even if they are defined.

Disable external aggregates
Disabling and enabling external aggregates is useful during the cube and application development phase to measure the impact of external aggregates.

To measure the impact of external aggregates you need to gather output twice. First, you gather output when external aggregates are enabled and then you gather output again when external aggregates are disabled. You use these two sets of output to determine the impact of external aggregates.

Percentage of members in a level referenced in a filter predicate
If no limit is required, type 0.

This value must be between 0 and 100.

This parameter applies to retrieving data associated with a set of members. If there is a greater percentage retrieved than what you specify in this field, then the SQL query that is generated retrieves measure values for all of the members at the level (a speculative pre-fetch of data).

6. Restart the dynamic cube to apply your changes.

Types of caches used by dynamic cubes
Several types of caches are available for dynamic cubes to allow improvements to query response times.

Result set cache
Result set cache is an intermediate storage of multidimensional expression language (MDX) query results. This cache is stored on disk in a binary format. The in-memory portion of the result set cache stores the queries and the associated
security profile. If an MDX query from the dynamic query mode server to the IBM Cognos Dynamic Cubes engine matches an entry in the result set cache and matches the security profile for the cache, then the result is read from the disk and the query is not run.

**Expression cache**

The MDX engine caches the results of various intermediate MDX set expressions that are keyed by the expression, its query context, and the security profile of the user. If the MDX engine encounters a set expression that was previously executed then it will retrieve the result set from the expression cache instead of calculating the set expression.

The expression cache helps to relieve the costs associated with the time and memory it takes to run set expressions.

**Data cache**

The MDX engine sends data queries to the Cognos Dynamic Cubes engine. The result of each query that is retrieved from the database (fact table), database aggregate tables, and in-memory aggregate cache is stored in the data cache.

Before sending any query to the database, the Cognos Dynamic Cubes engine scans the data cache for all the entries that are able to provide some or all of the required data without querying the database.

The data cache is also known as the query cache.

**Member cache**

This cache contains cube members that are loaded from the source relational data source. The member cache can be refreshed when appropriate, such as when the source data is changed. Refreshing the member cache updates the cube with the latest metadata.

**Aggregate cache**

Aggregate Advisor analyzes dynamic cubes and suggests aggregates that can improve cube performance. The aggregate cache contains pre-calculated values for aggregations that are suggested by Aggregate Advisor. The pre-calculated values are results of queries to the database.

**Aggregate tables**

Data can be summarized in a table known as an aggregate table. An aggregate table contains detail fact data that is aggregated at a higher level relative to one or more of the dimensions associated with the data. Using an aggregate table allows the use of pre-calculated data from a data warehouse and decreases the amount of data that is accessed from the data warehouse.

**Workload log for Aggregate Advisor**

Aggregate Advisor can analyze the underlying model in a dynamic cube data source and recommend which aggregates to create. Aggregate Advisor runs on the query service and can reference a workload log file. If you want Aggregate
Advisor to consider information from workload logs when making recommendations, then the workload log file must be enabled on the dynamic cube.

When enabled, the workload log file captures the information that represents user workload usage such as running reports. This log file allows Aggregate Advisor to suggest aggregates, in-database or in-memory, that correspond directly to the reports contained in the log file.

To enable the dynamic cube workload log file, use the **Enable workload logging** cube property. For more information about this property and information about specifying dynamic cube properties, see “Setting dynamic cube properties” on page 125.

For more information about using Aggregate Advisor, see *IBM Cognos Dynamic Query Analyzer User Guide*.

**Clearing the workload log**

Clearing the workload log removes all entries for a dynamic cube from this log. This is useful if you want to capture new information about report usage.

You can create and schedule query service tasks for clearing the workload. For more information see “Creating and scheduling query service administration tasks” on page 130.

You can also clear the workload manually. For more information, see “Starting and managing dynamic cubes” on page 120.

**Setting general properties for a dynamic cube**

You can view and edit general properties of an individual dynamic cube data source.

**Procedure**

1. In IBM Cognos Administration, on the Status tab, click Data Stores.
2. In the Scorecard section, select the Data Stores - (All) view.
3. For the dynamic cube that you want to modify, click the Actions drop-down menu and click **Set properties**.
4. On the **General** tab, view or change the following properties as required:
   - **Type** The type of property. For example, a Dynamic Cubes database, a Dispatcher, or a Namespace are all a type of property.
   - **Owner**
     - The owner of the entry. By default, the owner is the person who created the entry. When the owner no longer exists in the namespace, or is from a different namespace than the current user, the owner shows as **Unknown**.
     - If you have Set policy permissions, then you can click **Make me the owner** to become the owner of the entry.
   - **Contact**
     - The person responsible for the entry. Click **Set the contact** and then click **Select the contact** to set the contact for the entry or click **Enter an email address** to enter the contact email address.
Location
The location of the entry in the portal and its ID. Click View the search path, ID and URL to view the fully qualified location and the ID of the entry in the content store.
Entries are assigned a unique identification (ID) number.

Created
The date the entry was created.

Modified
The most recent date that the entry was modified.

Icon
The icon for the entry. Click Edit to specify a different icon.

Disable this entry
When selected, users that do not have write permission for this entry cannot access it. The entry is no longer visible in the portal.

If an entry is disabled and you have write permission to it, the disabled icon is displayed next to the entry.

Hide this entry
Select this property to hide reports, packages, pages, folders, jobs, and other entries. Hide an entry to prevent it from unnecessary use, or to organize your view. The hidden entry is still accessible to other entries. For example, a hidden report is accessible as a drill-through target.

A hidden entry remains visible, but its icon is faded. If you clear the Show hidden entries check box in my area options My Preferences, the entry disappears from your view.

You must have access to the Hide Entries capability granted by your administrator to see this property.

Language
A list of languages that are available for the entry name, screen tip, and description according to the configuration that was set up by your administrator.

Name
The name of the entry for the selected language.

Note: Renaming a dynamic query cube can cause problems for objects that reference this cube. For this reason, you should not change the name of the dynamic cube data source.

Screen tip
An optional description of the entry. The screen tip displays when you pause your pointer over the icon for the entry in the portal. Up to 100 characters can be used for a screen tip.

Description
An optional description of the entry, which displays in the portal when you set your preferences to use the details view.
Details view is displayed only in Public Folders and My Folders.

Access Account
The access account is used by the dynamic cube data source to access the relational database. The dynamic cube uses the data source signon credentials to access the relational database that contains the data.
warehouse of a dynamic cube. You can select which Cognos account to use based on its credentials. You must create the credentials before you define the access account.

For more information about defining the access account, see "Assigning the data access account for dynamic cubes" on page 115.

Creating and scheduling query service administration tasks

Administrators can create and schedule query service tasks for dynamic cube data sources. For example, you can schedule cache clearing, and clear the cache to control memory usage by a specific data source or cube.

The following query service tasks can be scheduled for one or more cubes:
• Clear workload log.
• Refresh data cache.
• Refresh member cache.
• Refresh security settings.
• Restart.
• Start.
• Start cube and source cubes.
• Stop after active tasks complete.
• Stop immediately.

You can create query service administration tasks and run them on demand. You can run the administration tasks at a scheduled time or based on a trigger, such as a database refresh or an email. You can schedule them as part of a job. You can also view the run history of query service administration tasks. For more information, see the IBM Cognos Business Intelligence Administration and Security Guide.

Before you begin

When you create and schedule tasks for dynamic cubes, you must schedule start and stop tasks for source cubes and virtual cubes separately. Consider the following factors when scheduling start and stop tasks for dynamic cubes:
• Source cubes that are part of a virtual cube must be scheduled to start first.
• If source cubes are part of a virtual cube, the virtual cube must be scheduled to stop before the source cubes.
• You need to provide enough time for source cubes to start before scheduling a virtual cube to start. The same condition applies when you schedule virtual and source cubes to stop.

To start virtual cubes, you can use the Start cube and source cubes action.

Procedure

1. In IBM Cognos Administration, on the Configuration tab, click Content Administration.
2. In the page toolbar, click the New Query service administration task icon, and then click Dynamic cube.
3. Specify a name, description, screen tip, and location for the new task, and click Next.
4. Select an operation.
For detailed information about the different actions, see “Starting and managing dynamic cubes” on page 120.

5. Select the required Server Group, Dispatcher, and Cubes, and click Next.

6. Choose how to run the task:
   - To run the task now or later, click Save and run once and click Finish. Specify a time and date for the run, and then click Run.
   - To schedule the task at a recurring time, click Save and schedule and click Finish. Then, select frequency and start and end dates.

   **Tip:** To temporarily disable the schedule, select the Disable the schedule check box.
   - To save the task without scheduling or running, click Save only and click Finish.

**Results**

After they are saved, the query service administration tasks appear on the Configuration tab, in Content Administration.

**What to do next**

You must delete a scheduled task if you delete the associated cube from the query service. Otherwise, your scheduled tasks will point to a cube that no longer exists.

**Setting access permissions for security views**

The model contains the security views that were defined for the dynamic cube in IBM Cognos Cube Designer. Administrators set access permissions for the security views.

**About this task**

Security views can be accessed from the model within a dynamic cube data source. A model view in IBM Cognos Administration is equivalent to a security view in Cognos Cube Designer.

By default, when a dynamic cube is published to the content store, the group Everyone has access to the model view. Administrators must override the access permissions to remove Everyone and add the appropriate users, groups, or roles to the model view.

Only read permissions are required to give the users, groups, or roles access to the metadata in a dynamic cube.

**Procedure**

1. In IBM Cognos Administration, on the Status tab, click Data Stores.

   In the Scorecard section, you see a list of all published dynamic cube data sources in the IBM Cognos BI environment.

2. Point to the data source that you want to edit, and from the Actions drop-down menu, click Edit security view permissions.

   The available security views are listed in the model.

3. For the selected security view, in the Actions column, click the Set properties icon.
4. Choose whether to use the permissions of the parent entry or specify permissions specifically for the entry:
   • To use the permissions of the parent entry, clear the **Override the access permissions acquired from the parent entry** check box, then click **OK** if you are prompted to use the parent permissions.
   • To set access permissions for the entry, select the **Override the access permissions acquired from the parent entry** check box, and proceed to step 5.

5. Optional: If you want to remove an entry from the list, select its check box and click **Remove**.

   **Tip:** If you want to select all entries, select the check box at the top of the list. Clear the check box to deselect all entries.

6. To specify the entries for which you want to grant or deny access, click **Add**, and choose how to select entries:
   • To choose from listed entries, click the appropriate namespace, and then select the check boxes next to the users, groups, or roles.
   • To search for entries, click **Search** and in the **Search string** box, type the phrase you want to search for. For search options, click **Edit**. Find and click the entry you want.
   • To type the name of entries that you want to add, click **Type** and type the names of groups, roles, or users using the following format, where a semicolon (;) separates each entry:
     
     namespace/group_name;namespace/role_name;namespace/user_name;
     
     Here is an example: Cognos/Authors;LDAP/scarter;

7. Click the right-arrow icon and when the selected entries appear in the **Selected entries** box, click **OK**.

   **Tip:** To remove entries from the **Selected entries** box, select them and click **Remove**. To select all entries in a list, click the check box in the upper-left corner of the list. To make the user entries visible, click **Show users in the list**.

8. Grant read permissions for each entry in the list, and click **OK**.

   **Tip:** In the **Permissions** column, an icon appears next to the user, group, or role. This icon represents the type of access granted or denied to the entry.

9. If you want to remove access permissions that were previously set for the child entries so that the child entries can acquire permissions set for this entry, in the **Option** section, select the **Delete the access permissions of all child entries** check box.

   This option appears only with entries that are containers. You can use it to restrict access to a hierarchy of entries. Select this option only when you are certain that changing access permissions of the child entries is safe.
Appendix A. Accessibility features

Accessibility features help users who have a physical disability, such as restricted mobility or limited vision, to use information technology products successfully.

The major accessibility features for IBM Cognos Cube Designer are described in the following list. You can
• customize the display to enhance accessibility. For example, you can enable a focus ring that emphasizes the selected element.
• use shortcut keys to navigate and trigger actions.
• apply operating system display settings, such as high-contrast display.

For more information about the commitment that IBM has to accessibility, see the IBM Accessibility Center (http://www.ibm.com/able).

Accessibility features in Cognos Cube Designer

You can customize the IBM Cognos Cube Designer display to enhance accessibility.

The View menu includes the following display controls.

Table 42. View menu options

<table>
<thead>
<tr>
<th>View menu item</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Show Access Keys</td>
<td>Adds a numeric identifier to each pane. To navigate to a different pane, press Alt+Shift+pane number. The navigation control works when Show Access Keys is disabled.</td>
</tr>
<tr>
<td>Show Focus Rectangle</td>
<td>Displays a dotted rectangle around the object that has the current keyboard focus.</td>
</tr>
</tbody>
</table>

Keyboard shortcuts for Cognos Cube Designer

You can use keyboard shortcuts to navigate through and perform some tasks in IBM Cognos Cube Designer.

Table 43. Keyboard shortcuts for Cognos Cube Designer

<table>
<thead>
<tr>
<th>Applies to</th>
<th>Description</th>
<th>Keyboard shortcut</th>
</tr>
</thead>
<tbody>
<tr>
<td>General</td>
<td>Perform the default action for an active command button.</td>
<td>Enter or Spacebar</td>
</tr>
<tr>
<td>General controls</td>
<td>Move forward to the next control at the same level.</td>
<td>Tab</td>
</tr>
<tr>
<td>General controls</td>
<td>Move backward to the previous control at the same level.</td>
<td>Shift+Tab</td>
</tr>
<tr>
<td>Applies to</td>
<td>Description</td>
<td>Keyboard shortcut</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
<td>-------------------</td>
</tr>
<tr>
<td>Check boxes</td>
<td>Toggle a check box from selected to cleared or cleared to selected. (Tip: This shortcut also applies to other settings that can have an on or off state.)</td>
<td>Spacebar</td>
</tr>
<tr>
<td>Radio buttons that are not in a group</td>
<td>Move to the next radio button and select it.</td>
<td>Tab</td>
</tr>
<tr>
<td>Radio button groups</td>
<td>Move to the next radio button in the group and select it.</td>
<td>Right arrow</td>
</tr>
<tr>
<td></td>
<td>Move to the previous radio button in the group and select it.</td>
<td>Left arrow</td>
</tr>
<tr>
<td>Drop-down lists</td>
<td>Open and display the drop-down list contents.</td>
<td>Alt+Down arrow</td>
</tr>
<tr>
<td>Drop-down lists</td>
<td>Close an open drop-down list.</td>
<td>Alt+Up arrow</td>
</tr>
<tr>
<td>Tree controls</td>
<td>Move to the first selectable node below, or, if the node below has child nodes and the node is expanded, move to the first child node.</td>
<td>Down arrow</td>
</tr>
<tr>
<td>Tree controls</td>
<td>Move to the first selectable node above.</td>
<td>Up arrow</td>
</tr>
<tr>
<td>Tree controls</td>
<td>Expand the selected node or move to the first selectable child node.</td>
<td>Right arrow</td>
</tr>
<tr>
<td>Tree controls</td>
<td>Collapse the selected node, move to the parent node, or move to the first selectable node above.</td>
<td>Left arrow</td>
</tr>
<tr>
<td>Tree controls</td>
<td>Move to the first node in a tree control.</td>
<td>Home</td>
</tr>
<tr>
<td>Tree controls</td>
<td>Move to the last node in a tree control.</td>
<td>End</td>
</tr>
<tr>
<td>Menus</td>
<td>Move to the next available menu item.</td>
<td>Down arrow</td>
</tr>
<tr>
<td>Applies to</td>
<td>Description</td>
<td>Keyboard shortcut</td>
</tr>
<tr>
<td>-----------------</td>
<td>------------------------------------------</td>
<td>------------------</td>
</tr>
<tr>
<td>Menus</td>
<td>Move to the previous available menu item.</td>
<td>Up arrow</td>
</tr>
<tr>
<td>Menus</td>
<td>Expand the child menu items.</td>
<td>Right arrow</td>
</tr>
<tr>
<td>Menus</td>
<td>Collapse the child menu items.</td>
<td>Left arrow</td>
</tr>
<tr>
<td>Context menus</td>
<td>Open the context menu for the selected item.</td>
<td>Shift+F10</td>
</tr>
<tr>
<td>Context menus</td>
<td>Close an open context menu.</td>
<td>Esc</td>
</tr>
<tr>
<td>Scrolling</td>
<td>Scroll down.</td>
<td>Down arrow</td>
</tr>
<tr>
<td>Scrolling</td>
<td>Scroll up.</td>
<td>Up arrow</td>
</tr>
<tr>
<td>Columns</td>
<td>Change the width.</td>
<td>Ctrl+Shift+►</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Ctrl+Shift+◄</td>
</tr>
</tbody>
</table>
Appendix B. Report considerations

There are a number of points to consider when you view report data based on a published dynamic cube.

Calculated members in reports

For most reports, IBM Cognos Dynamic Cubes calculated members are used the same way as regular members. However, because of some different constraints and capabilities, the report user may encounter unexpected results. In these cases, you must consider the required type and behavior of the members to obtain the desired output. In reporting environments, calculated members appear to be identical to regular members. It is a good practice to use a naming convention so that report users can easily identify calculated members.

The values of calculated members and measures are not retained within a dynamic cube. The values are computed at every occurrence within reports and analyses when executed.

You create Cognos Dynamic Cubes calculated members manually. Cognos Dynamic Cubes relative time calculated members are specialized calculated members automatically added to a relative time hierarchy and cannot be modified.

Calculated members that you manually create have the following characteristics:
- Each occurrence of a single calculated member in a report or analysis is considered unique. (SET operations, Filtering calculated members)
- They do not have siblings or children.
- They should not be nested.
- Their rank value in IBM Cognos Analysis Studio is always Null.

Relative time calculated members

The relative time feature generates three types of calculated members.

The Period to Date Change and Period to Date Growth relative time calculated members share the following characteristics with Cognos Dynamic Cubes calculated members.
- Each occurrence of a single calculated member in a report or analysis is considered unique. (SET operations, Filtering calculated members)
- They do not have siblings or children.
- They should not be nested.
- Their rank value in IBM Cognos Analysis Studio is always Null.

The Current Period, Prior Period, Current Period to Date and Prior Period to Date members may have children. Therefore, the functions CHILDREN, DESCENDANT, FIRSTCHILD and LASTCHILD can return results. These relative time calculated members share the following characteristics with Cognos Dynamic Cubes calculated members:
- Each occurrence of a single calculated member in a report or analysis is considered unique. (SET operations, Filtering calculated members)
• They should not be nested
• Their rank value in IBM Cognos Analysis Studio is always Null.

Reference relative time members refer to other members within the time hierarchy and have the same caption and member key values as the members to which they refer. Within the context of other reference members, these members behave the same as Cognos Dynamic Cubes calculated members. Unlike Cognos Dynamic Cubes calculated members, these members are not considered unique, they can have children and they can be nested. Reference members at the same level are siblings of other reference members. When applied to a reference member, functions such as FIRSTSIBLING or NEXTMEMBER will return a reference member. Their rank value in IBM Cognos Analysis Studio is always Null.

**SET operations**

Because a calculated member is considered to be unique from all other calculated members, the UNION, EXCEPT, UNIQUE, and INTERSECT functions may give results that appear incorrect.

In the following examples, [USA] and [Canada] are regular members and [CM1] and [CM2] are calculated members.

*Table 44. Examples of SET operations with calculated members*

<table>
<thead>
<tr>
<th>Example</th>
<th>Result Set</th>
</tr>
</thead>
<tbody>
<tr>
<td>UNION</td>
<td>SET</td>
</tr>
<tr>
<td>( SET( [USA], [CM1], [CM2] ), SET( [USA], [Canada], [CM1] )</td>
<td>( [USA], [CM1], [CM2], [Canada], [CM1] )</td>
</tr>
<tr>
<td></td>
<td>The member [CM1] appears twice in the result.</td>
</tr>
<tr>
<td>EXCEPT</td>
<td>SET</td>
</tr>
<tr>
<td>( SET( [USA], [CM1], [CM2] ), SET( [USA], [Canada], [CM1] )</td>
<td>( [CM1], [CM2], [Canada], [CM1] )</td>
</tr>
<tr>
<td></td>
<td>The member [USA] is removed, but the member [CM1] appears twice in the result.</td>
</tr>
<tr>
<td>UNIQUE</td>
<td>SET</td>
</tr>
<tr>
<td>( SET( [USA], [CM1], [USA], [CM1], [Canada] )</td>
<td>( [USA], [CM1], [CM1], [Canada] )</td>
</tr>
<tr>
<td></td>
<td>The member [CM1] appears twice in the result.</td>
</tr>
<tr>
<td>INTERSECT</td>
<td>SET</td>
</tr>
<tr>
<td>( SET( [USA], [CM1], [CM2] ), SET( [USA], [Canada], [CM1] )</td>
<td>( [USA] )</td>
</tr>
<tr>
<td></td>
<td>Calculated members do not appear in the intersection of two sets.</td>
</tr>
</tbody>
</table>

**Filtering calculated members**

Since calculated members are considered to be unique from all other calculated members, a filter will not remove the members.

If a report contains a filter based on IBM Cognos Dynamic Cubes calculated members and the same hierarchy is visible in the report, the data values in the report will be correct. However, the filter will not remove visible members from the report. If the same hierarchy is not visible in the report, the report output will be as expected.
Nesting calculated members

IBM Cognos Dynamic Cubes calculated members should not be nested. Because all calculated members are considered unique, the dynamic query mode query planner resolves the intersection to an empty set. The rows remain in the reports but the values are Null.

Calculated members siblings and children

IBM Cognos Dynamic Cubes calculated members do not have siblings or children. Functions that require a member sibling or child as a result will always be Null.

- NEXTMEMBER( [CM1] ) = NULL
- PREVMEMBER( [CM2] ) = NULL
- LEAD( [CM1], 0 ) = NULL
- LAG( [CM2], 0 ) = NULL

Cognos Analysis Studio rank

In IBM Cognos Analysis Studio, the rank of an IBM Cognos Dynamic Cubes calculated member is always Null. The context in which the rank is computed and the context used to compute the values visible in the cross tab are not the same. Because, the computed rank values could contradict the visible values, the rank is always set to Null.

Relative time calculated members in reports

IBM Cognos Dynamic Cubes relative time members are specialized calculated members that are added to a time hierarchy.

The relative time feature generates three types of calculated members

Period to Date Change, Period to Date Growth

These relative time calculated members share the following characteristics with Cognos Dynamic Cubes calculated members.

- They are considered unique.
- They do not have siblings or children
- They should not be nested
- Their rank value in IBM Cognos Analysis Studio is always null.

Current Period, Prior Period, Current Period to Date, Prior Period to Date

These members behave the same as Cognos Dynamic Cubes calculated members with one exception. These members may have children. Therefore, the functions CHILDREN, DESCENDANT, FIRSTCHILD and LASTCHILD can return results.

These relative time calculated members share the following characteristics with Cognos Dynamic Cubes calculated members:

- They are considered unique.
- They should not be nested
- Their rank value in IBM Cognos Analysis Studio is always null.
Reference relative time members

These members refer to other members within the time hierarchy and have the same caption and member key values as the members to which they refer. Within the context of other reference members, these members behave the same as Cognos Dynamic Cubes calculated members with one exception. These members can have children. Reference members at the same level are siblings of other reference members. When applied to a reference member, functions such as FIRSTSIBLING or NEXTMEMBER will return a reference member.

These relative time calculated members share the following characteristics with Cognos Dynamic Cubes calculated members:

- They are considered unique.
- They can be nested
- Their rank value in IBM Cognos Analysis Studio is always null.

Removal of padding members from reports

The use of padding members can result in skewed calculations related to the members of a hierarchy level. If a level contains padding members, they are included in the count of members. In addition, because padding members can have associated fact data values, this can skew the value of aggregates computed on a level-basis.

For example, in a State/City hierarchy, if the state of California has no city level members, a padding member at the city level is created as a child of California to balance the hierarchy. If the Sales measure value for California is 100, then the child padding member also has a value of 100. The number of city entries across all states is now inflated by 1 and that the sum of all Sales values across all cities is inflated by 100.

To remove skewed data from a report, you can define a filter for a set of members based on a dynamic cube.

Hierarchies with padding members are not displayed as ragged or unbalanced in the IBM Cognos studios. A report user can identify ragged and unbalanced hierarchies by looking for members with a blank caption or the same caption as their parent. These members have a NULL business key because they do not represent real members. Filtering members with a NULL business key removes all the padding members. A report filter such as FILTER( MEMBERS( [My Level] ), [My Level].[My Level - Key] = NULL ) removes padding members from the report.
Appendix C. Troubleshooting

This section provides solutions for problems you may encounter when using IBM Cognos Dynamic Cubes.

Possible overflow in measure attributes

Measure attributes in a dynamic cube may be too small to hold aggregate values of the measures.

The Measure properties of **Data type**, **Precision** and **Scale** are inherited from the relational database metadata and cannot be modified. If the aggregate value of a measure exceeds the size of the attribute, you see an error indicating that an overflow has occurred. For example, a Quantity Measure defined as Int(4) overflows when summed in a dynamic cube.

To avoid overflow errors, first evaluate the database columns you want to use as measures. If the resulting data type will not accommodate the aggregation value of the measure, do the following:

- Create a default measure for the database column you want to use as a measure.
- Evaluate the measure to determine an appropriate aggregate size.
- Hide the original measure that you determined to cause an overflow.
- Create a new measure.
- Define the measure using the expression property. The expression must be an explicit cast of the original measure into a larger data type.

The syntax for the CAST function is `CAST (<expression>, <datatype>)`

For example:

```
CAST( [MyDataItem], varchar(10))
```

If casting to a data type that accepts size, precision, or scale, those parameters appear in parentheses after the data type. For example

```
CAST( [MyDataItem], decimal(10,2))
```

In-memory aggregates fail to load

If in-memory aggregates fail to load when a dynamic cube starts, additional memory may be required for the aggregate cache.

In-memory aggregates are defined by running Aggregate Advisor in IBM Cognos Dynamic Query Analyzer and saving the in-memory aggregate definitions. When a dynamic cube is restarted, the in-memory aggregates are loaded. If they fail to load, check the dynamic cube error log for the following message:

"Loading of in-memory aggregates was skipped because the value for the 'Maximum amount of memory to use for aggregate cache' property is zero. To enable loading in-memory aggregates, update the property to a value greater than zero to be the amount of memory to allocate for the aggregate cache."
In IBM Cognos Administration, open the properties for the cube and set the **Maximum amount of memory to use for aggregate cache** to a value greater than or equal to the one used when creating the recommendations in Aggregate Advisor.

**Issues with dynamic cubes that contains members with duplicate level keys**

In the previous release, it was possible to model a hierarchy level with members that contain duplicate level keys.

In IBM Cognos Dynamic Cubes 10.2.1, when you browse members with duplicate level keys in IBM Cognos Cube Designer, it constructs the member tree as you expand each member, and does not check for members with duplicate level keys. However, when you start a dynamic cube that contains members with duplicate level keys, it might now fail with an error. This is a result of improved validation in this release.

To overcome this error, you can update the hierarchy level that contains duplicate level key members by specifying additional columns to ensure that the level key is unique.

If you do not want to update the dynamic cube, you can disable the new validation check by setting a parameter named `disableDuplicateLevelCheck` to `true`. For more information about setting parameters, see the *IBM Cognos Business Intelligence Administration and Security Guide*. 
Notices

This information was developed for products and services offered worldwide.

IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service. This document may describe products, services, or features that are not included in the Program or license entitlement that you have purchased.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not grant you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Licensing
IBM Corporation
North Castle Drive
Armonk, NY 10504-1785
U.S.A.

For license inquiries regarding double-byte (DBCS) information, contact the IBM Intellectual Property Department in your country or send inquiries, in writing, to:

Intellectual Property Licensing
Legal and Intellectual Property Law
IBM Japan Ltd.
19-21, Nihonbash-Hakozakicho, Chuo-ku
Tokyo 103-8510, Japan

The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law: INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Any references in this information to non-IBM Web sites are provided for convenience only and do not in any manner serve as an endorsement of those Web sites. The materials at those Web sites are not part of the materials for this IBM product and use of those Web sites is at your own risk.
IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you.

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact:

IBM Software Group
Attention: Licensing
3755 Riverside Dr
Ottawa, ON K1V 1B7
Canada

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

The licensed program described in this document and all licensed material available for it are provided by IBM under terms of the IBM Customer Agreement, IBM International Program License Agreement or any equivalent agreement between us.

Any performance data contained herein was determined in a controlled environment. Therefore, the results obtained in other operating environments may vary significantly. Some measurements may have been made on development-level systems and there is no guarantee that these measurements will be the same on generally available systems. Furthermore, some measurements may have been estimated through extrapolation. Actual results may vary. Users of this document should verify the applicable data for their specific environment.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

All statements regarding IBM's future direction or intent are subject to change or withdrawal without notice, and represent goals and objectives only.

This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to the names and addresses used by an actual business enterprise is entirely coincidental.

If you are viewing this information softcopy, the photographs and color illustrations may not appear.
Trademarks

IBM, the IBM logo and ibm.com are trademarks or registered trademarks of International Business Machines Corp., registered in many jurisdictions worldwide. Other product and service names might be trademarks of IBM or other companies. A current list of IBM trademarks is available on the Web at "Copyright and trademark information" at www.ibm.com/legal/copytrade.shtml.

The following terms are trademarks or registered trademarks of other companies:

• Java and all Java-based trademarks and logos are trademarks or registered trademarks of Oracle and/or its affiliates.
Index

A
accessibility features 133
keyboard shortcuts 133
adding to query service 119
administration 113
Aggregate Advisor 128
in-database 37
in-memory 37
aggregate awareness 36, 85
aggregate cubes
  defining 88
  defining automatically 87
  defining manually 87
  filtering data for 89
  in-database aggregates 36
  in-memory aggregates 36
  modeling 85
  overview 36
  parent-child dimension 88
  properties 85
aggregate slicer 89
aggregation rules
  calculated 28, 29
  defining 66
  time state 29
attributes
  level key 22, 23
  level unique key 23
  member caption 23
  member description 23
  overview 23
  relational mapping 25
  support multiple locales 82

B
balanced hierarchies 16
browsing
  members in a hierarchy 58
business keys
  See level keys

C
calculated measures 71, 137
calculated members 71, 79, 137, 139
defining 76
  relative time 137, 139
  security 105
unique 137
Cognos Cube Designer
  overview 39
  user interface 39
Cognos Dynamic Cubes
  overview 5
  workflow 9
configuring
  dynamic cubes 69, 117

Content Manager data source
  importing metadata 42
creating
  measure folders 68
creating a signon
dynamic cubes 116
current data 35
current period
  expression examples 78
current period examples
  expressions 78
current period expressions examples 78

D
data members
  hiding 20
  leaf 20
  non-leaf 20
  overview 20
  parent-child hierarchies 20
  showing 20
data source signon 116
defining
  dimension filters 59
dimensions 47, 49
  level unique keys 54
  measure dimension filters 68
  security filters 108
  security views 110
  sort order of members 55
  virtual dimensions 93
denying
  security 101, 109
deploying
dynamic cubes 69
design language 82
dimension filters
defining 59
dimensional metadata
  attributes 23
dimensions 15
  hierarchies 15
  joins 23
  levels 22
  modeling 47
  overview 15
  parent-child hierarchies 20
dimensions
  defining 47, 49
degenerate 15
  filter 58
  modeling 47
  parent-child 15
  regular 15
  relational mapping 25
  relative time 76, 78
dynamic cubes
  administration 113
  advanced modeling 71
dynamic cubes (continued)
    based on a relational table 62
    browse member issue 142
    configuration 128
    configuring 69, 117
    creating a signon 116
    creating from InfoSphere Warehouse Cubing Services
      model 43
    defining manually 61
    deploying 69
    editing 45
    errors 45
    failure to start 142
    generating from a relational table 61
    importing metadata for 41, 42
    issues 45
    measure dimension 24
    measures 27
    modeling 61
    multiple dispatchers 117
    opening 45
    overview 24
    publishing 69
    saving 45
    security 101
    specify a server group name 117
    starting 69
    starting and managing 120
    validating 45
    warnings 45
    dynamic cubes removing from query service
      adding to query service 119

E
    editing
      dynamic cubes 45
      projects 45
    errors 45
    examples
      24 month rolling window 79
      calculated members 72
      level current period expressions 78
      relative time 79
    execution trace
      query service 122
    expression editor
      defining calculated members 76
      extraneous padding members
        hiding 19
        showing 19

F
    features
      new 1
    filtering data
      aggregate cubes 89
    filters
      dimension 58, 59
      measure dimension 67, 68
    folders
      measure 68

G
    granting
      security 101, 109

H
    hiding
      data members 20
      extraneous padding members 19
      measures 65, 71
    hierarchies
      balanced 16
      browsing members 58
      default 104
      defining 51
      defining security for 101
      modeling 50
      multiple 16
      overview 15
      padding members 18
      ragged 17
      relational mapping 25
      security 109
      sort order of members 55
      unbalanced 16

I
    idle connection timeout
      query service 122
    importing
      Content Manager data source metadata 42
      InfoSphere Warehouse Cubing Services model 43
      metadata 41
      in-database 37
      in-memory 37
      in-memory aggregates
        fail to load error 141
    InfoSphere Warehouse Cubing Services
      importing cube metadata 43
      issues 45

J
    joins
      measure-to-dimension 61, 67
      overview 23
      relational mapping 26

K
    keyboard shortcuts 133

L
    languages
      adding to metadata objects 82
      design language 82
      support multiple locales 82
      supporting different locales 82
      level keys 22, 54
      attribute 23
      level unique keys 22
      attribute 23
level unique keys (continued)
  defining 54
levels
  alternative modeling 22
  best practice modeling 22
  modeling 51
  overview 22
  relational mapping 25
locales
  adding languages to metadata objects 82
  adding to members and attributes 82
  design language 82
  supporting 82

M
  measure dimension 24
    filter 67
  measure dimension filters
    defining 68
  measure folders
    creating 68
    sorting 69
  measure-to-dimension joins
    defining 67
  measures
    aggregation rules 28, 29
    calculated 71, 137
    calculated measures 27
    dynamic calculated measures 27
    folders 68
    hidden 65, 71
    modeling 64, 65, 66
    overview 27
    regular aggregation 28, 29
    security 109
    sorting 69
    visible 65, 71
  member caption
    attribute 23
  member description
    attribute 23
members
  browsing in hierarchies 58
  calculated 71, 137, 139
  defining security for 101
  defining sort order 55
  leaf 20
  non-leaf 20
  predefined relative time 78
  support multiple locales 82
metadata
  adding languages 82
  importing 41
  importing from a Content Manager data source 42
  importing metadata for 43
  modeling
    advanced 71
    aggregate cubes 85
    dimensional metadata 47
    dimensions 47
    dynamic cubes 61
    hierarchies 50
    levels 51
    measures 64
    parent-child hierarchies 55
    relative time dimensions 76
modeling (continued)
  virtual cubes 91
  virtual dimensions 92
  virtual hierarchies 93
  virtual measures 97
  virtual members 95
  multiple hierarchies 16

N
  new features 1
    administration interface 3
    Aggregation Advisor 2
    aggregation rules 2
    dimension filters 1
    embedded prompts and macros 1
    generate cube 2
    generate dimension 2
    measure dimension filters 1
    measure folders 1
    measure sorting 1
    metadata import 2
    performance issues 3
    security 2
    non-rollup hierarchy 20, 21

O
  objects
    errors 45
    issues 45
    validating 45
    warnings 45
  opening
    dynamic cubes 45
    projects 45
  overview
    aggregate cubes 36
    Cognos Cube Designer 39
    Cognos Dynamic Cubes 5
    dimensional metadata 15
    dimensions 15
    dynamic cubes 24
    hierarchies 15
    parent-child hierarchies 20
    user interface 39
    virtual cubes 32

P
  padding members 18
  removing from reports 140
  parent member 71
  parent-child hierarchies
    data members 20
    defining 57
    modeling 55
    overview 20
    performance issues 3
    pre-cached historical data 35
    projects
      adding support for locales 82
      defining virtual cubes 91
      design language 82
      editing 45
      errors 45
projects (continued)
  importing metadata for 42, 43
  issues 45
  opening 45
  saving 45
  validating 45
  warnings 45
properties
  query service 122
publishing
  dynamic cubes 69

Q
query planning trace
  query service 122
query service
  creating and scheduling administration tasks 130
  execution trace 122
  properties 122
  query planning trace 122
  run tree trace 122
  starting 125
  stopping 125
  write model to file 122

R
ragged hierarchies 17
regular aggregation 28, 29
relational tables
  using for dynamic cubes 62
relative time
  calculated members 137, 139
relative time dimensions
  defining 78
  modeling 76
renewing
  trusted credentials 116
role-based 108
rollup hierarchy 19, 21
run tree trace
  query service 122

S
saving
  dynamic cubes 45
  projects 45
scenarios
  virtual cube 35
schedules
  query service administration tasks 130
schemas
  snowflake 24
  star 24
security
  calculated members 105
  denying access 101, 109
  dynamic cubes 101
  granting access 101, 109
  hierarchies 101, 109
  measures 109
  members 101
  tuple 110
security (continued)
  virtual cubes 101
  security filters 101
  defining manually 108
  security views 101, 109
  defining 110
  shared dimensions 35
  showing
    data members 20
    extraneous padding members 19
  simple aggregate table 87
  snowflake schemas 24
  sort order
    defining for members 55
  sorting
    measure folders 69
    measures 69
    star schemas 24
    starting
      dynamic cubes 69
      query service 125
      starting and managing
dynamic cubes 120
  virtual cubes 120
  stopping
    query service 125

T
troubleshooting
  in-memory aggregates 141
  members with duplicate level keys 142
trusted credentials
  renewing 116
tuple
  security 110

U
unbalanced hierarchies 16
user interface 39

V
viewing
  members in a hierarchy 58
virtual cubes
  current data 35
  defining in a project 91
  defining virtual dimensions 93
  modeling 91
  objects 32
  overview 32
  pre-cached historical data 35
  scenarios 35
  security 101
  shared dimensions 35
  starting and managing
dynamic cubes 120
  virtual cubes 120
  virtual dimensions 32
  defining 93
  modeling 92
virtual hierarchies
  modeling 93
virtual measures 32
  modeling 97
  virtual members 32
virtual members (continued)
  modeling 95
visible 71

W
warnings 45

*workflow*
  Cognos Dynamic Cubes 9
workload logs 128
write model to file
  dynamic cubes
    query service properties 122
    query service 122