IBM Cognos Software Development Kit
Version 12.0.x

Framework Manager Developer Guide

©

Product Information

This document applies to IBM Cognos Analytics version 12.0.0 and may also apply to subsequent releases.

Copyright

Contents

Introduction

IBM® Cognos® Framework Manager is a data modeling product. It lets users import metadata from one or more data sources and transform it into a business-oriented model for creating reports.

This guide is for developers interested in using the collection of cross-platform Web services, libraries, and programming interfaces provided with the IBM Cognos Software Development Kit, to access the full functionality of Framework Manager. You can use the Framework Manager API to model metadata and publish packages without the use of the Framework Manager application.

The document includes both task-oriented and reference information, to help you implement custom solutions for metadata modeling.

Conceptual and procedural information is presented in the initial chapters. Background and reference information appears in the appendixes.

Audience

To use this guide effectively, you should be familiar with the following:

- Framework Manager
- XML, HTML, WSDL, and SOAP 1.1 coding standards
- XSL style sheets
- Authenticating users

Finding information

To find product documentation on the web, including all translated documentation, access IBM Knowledge Center (http://www.ibm.com/support/knowledgecenter).

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.

Accessibility features

Consult the documentation for the tools that you use to develop applications to determine their accessibility level. These tools are not a part of this product.

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

Chapter 1. The Framework Manager API

The Framework Manager API provides a platform-independent automation modeling interface. This interface has Framework Manager services and components that are linked through the BI Bus API. Communication on the BI Bus API consists of requests and responses in the form of standard Simple Object Access Protocol (SOAP) messages.

To learn more about the Framework Manager API, we recommend that you be familiar with the Framework Manager application. The Framework Manager application records all the actions (see "Actions" on page 8) you do that modify the metadata model. These actions are recorded in action logs (see "Action logs" on page 7). Use the Framework Manager application to perform the modeling tasks you need, and review the log file to see the results.

After you are familiar with the structure of action logs, you can create your own action logs to accomplish similar goals. You can use the Framework Manager API to model metadata and publish packages without the use of the Framework Manager application.

For information about the Framework Manager application and concepts, see the *IBM Cognos Framework Manager User Guide*. We also recommend that you read the BI Bus API and content management chapters of the *IBM Cognos Software Development Kit Developer Guide*.

Using the Framework Manager API

You can use the Framework Manager API to perform all the same metadata modeling tasks and processes as the Framework Manager application. For example, you can perform the following tasks:

- Import a data source.
- Enhance query subjects with SQL, expressions and filters.
- Create model query subjects to extend value of data source query subjects.
- Create a basic package.
- Publish a package to report authors.

The Framework Manager API supports two methods of modeling metadata: the ScriptPlayer and the Metadata Service. Both of these methods use action logs. The BmtScriptPlayer is a stand-alone command line utility capable of playing action logs. When you use the Metadata Service, you send requests through the BI Bus API. You can obtain requests from action logs.

An action is a request that is sent to the IBM Cognos Analytics server. Actions can be grouped together to perform certain modeling activities. Actions are recorded as elements of an XML document. This document is called an action log. For more information, see "Action logs" on page 7

The following Framework Manager application functionality is not supported on UNIX operating systems:

- Import of third-party metadata sources.
- Import of Architect, Impromptu, or DecisionStream XML files.
- Export of the Framework Manager model to Common Warehouse Metamodel (CWM) format.

Reference material

This guide includes reference material that you can use to create actions and transactions that either the Script Player or the Metadata Service can use.

- Chapter 3, "Model schema reference," on page 29
- Chapter 4, "Metadata Provider (Wrapper) reference," on page 115

- Chapter 5, “Custom properties for SAP BW,” on page 119

The reference information can assist you in adapting the API to your own purposes. Once you understand the basics, you can integrate the modeling framework with your other applications, regardless of the operating systems, platforms, and programming languages used to create them. As you gain expertise, you can use the API to customize the Framework Manager modeling tools to meet your own business needs.

The Model schema

The Model schema validates the `model.xml` file, the xml representation of the model. The Model Schema reference contains information about the elements and attributes in the `model.xml` file.

The Metadata Service

BI Bus API messages are XML documents encapsulated as SOAP requests that use the HTTP transport protocol.

The client wraps each transaction in a SOAP envelope so that it can be understood by the BI Bus API. The SOAP envelope contains a SOAP header, and SOAP body. The Metadata Service request, represented by an mdprovider element, is contained in the body of a SOAP request.

For each SOAP request, a response or fault is returned.

Generic requests create, open, save or close the model. Action requests modify the metadata or publish a package. Chapter 4, “Metadata Provider (Wrapper) reference,” on page 115 provides descriptions for each type of request.

Here is an example of a Metadata Service request. In this example, you create a parameter map named New_Parameter_Map:

```
<mdprovider type=action action=execute model=../temp/myproject/myproject.cpf>
<transaction seq=5 timestamp=20030221155630569>
  <action
    seq=1 type=Create>
     <inputparams>
      <param seq=1 type=integer>
         <value>1</value></param>
      <param seq=2 type=handle>
         <value>[].[parameterMaps]</value></param>
      <param seq=3 type=i18nstring>
         <value>New_Parameter_Map</value></param>
      <param seq=4 type=integer>
         <value>1</value></param>
     </inputparams>
    <domchanges/>
    <result success=t><outputparams/></result>
  </action>
</transaction>
</mdprovider>
```

Custom properties for SAP BW

When you create a Framework Manager model that is based on an SAP BW data source, information specific to SAP BW is stored in custom properties. The custom properties reference describes the objects that are required in your model, the properties that apply to them, and the descriptions and restrictions that apply to those properties.

For more information about metadata modeling based on an SAP BW data source, see the *IBM Cognos Framework Manager User Guide*.

For more information, see Chapter 5, “Custom properties for SAP BW,” on page 119.

Use Framework Manager to view action logs

The Framework Manager application records all the actions you do that modify the metadata model. These actions are recorded in action logs. Action logs are XML files that you can re-use and re-run in the

Framework Manager application. You can use these Framework Manager action logs as examples to help you create your own action logs for the API.

To view the action logs that represent modeling tasks performed, click **View Transaction History** in the **Projects** menu of the Framework Manager application. By default, the dialog box shows the `log.xml` file which contains all the transactions that have been run and saved in the project. This file is created the first time you save the project and exists until you delete the project.

To create an action log from the View Transaction dialog box, click the transactions that you wish to save and click **Save As Script**. You can create action logs that contain specific transactions or a single transaction.

You can then locate and examine the files to see what actions and sequence of actions that will be performed on the objects in the model.

Running action logs

There are two ways of running action logs. You can use the ScriptPlayer application or the Metadata Service.

ScriptPlayer

At the command prompt, navigate to the installation location of the `BmtScriptPlayer.exe`.

Use the following syntax to run the Script Player:

```
BmtScriptPlayer [-c|-m] <projectname> [-a <actionlogname>][options]
```

where *<projectname>* is the name of the project and *<actionlogname>* is the name of the action log.

For example,

```
BmtScriptPlayer -m goSales.cpf -a import.xml
```

Options

You can specify how the Script Player runs using the following options.

If you are working in a UNIX environment, you may want to create a script to hide credentials that are passed on the command line.

-a *FILEPATH*

Apply the specified action log.

FILEPATH is the path, including the file name, to the action log file.

-b NUM

Execute transactions with sequence number equal to or higher than the number specified by NUM.

The default is the first transaction.

-c *FILEPATH*

Create a new project.

FILEPATH is the path, including the file name, to the models project (.cpf) file.

Using this option without specifying an action log results in the creation of an empty model.

If the model specified in the *FILEPATH* already exists, it is silently replaced.

-e NUM

Execute transactions with sequence number equal to or lower than the number specified by NUM.

If the option is not specified, execution ends at the transaction with the highest sequence number or transaction number 9999, whichever comes first. For action logs that contain transactions with sequence numbers 10,000 and higher, this option must be used.

-g

Upgrade the model (if required).

If this option is not specified and the model was created with a previous version, execution terminates.

If you specify this option without specifying an action log, only the model upgrade is performed.

-h

Specify the elapsed time in milliseconds. This option is not available on Unix or Linux® operating systems.

-i *FILEPATH*

Specify the path, including the file name, to a file that contains the authentication information.

-k *DIRECTORY*

Specify the install directory.

-l *FILEPATH*

Specify the path, including the file name, to a file that contains the options to be used when running Script Player.

-L

Specify that the object ID logs (`IDLog.xml`) not be created. This option is not available on Unix or Linux operating systems.

-m *FILEPATH*

Open an existing project.

FILEPATH is the path, including the file name, to the models project (.cpf) file.

-n

Do not save the model.

This option can be used to test action log files.

-p *PASSWORD*

Authenticate using the specified password (if required).

-s *NAMESPACE*

Authenticate using the specified namespace (if required).

-t *DIRECTORY*

Specify the template directory.

-T *PASSWORD*

Specify a security passport. A passport is an encrypted string used to allow secure conversations for the plug-ins that need it.

-u *USER*

Authenticate using the specified user name (if required).

-x

Terminate the test run when there is a transaction error.

By default, the script player only terminates with severe errors such as an invalid model or action log, and continues executing, even if some minor transactions fail.

-y *PASSPORT*

Authenticate using the specified passport (if required).

This option overrides other specified credentials (-s, -p, and -u). The Script Player skips authentication and associates the specified passport with the session.

-z
Create extended action log with action output parameters..

Examples

This table shows some examples of Script Player commands.

Table 1. Script Player commands examples

Command	Description
`BmtScriptPlayer -c <projectname>`	Create a project.
`BmtScriptPlayer -c <projectname> -a <actionlogname>`	Create a project and apply all the transactions from the action log.
`BmtScriptPlayer -c <projectname> -a <actionlogname> -b2 -e20`	Create a project and apply the transactions numbered 2-20 from the action log.
`BmtScriptPlayer -m <projectname> -a <actionlogname> -e20`	Open an existing project and apply the transactions numbered 1-20 from the action log.
`BmtScriptPlayer-m <projectname> -a <actionlogname> -n`	Open an existing project and apply all the transactions from the action log. Do not save the project.

Example - run the script player

You must install Framework Manager and some sample action logs to run this example.

The script player, `BmtScriptPlayer.exe`, is available at `installation_location\bin`. To install the sample action logs, go to the IBM Accelerator Catalog (https://accelerator.ca.analytics.ibm.com/bi/?perspective=authoring&pathRef=.public_folders%2FIBM%2BAccelerator%2BCatalog%2FContent%2FTOL00018) and then click **Download**. Extract the downloaded files to `installation_location\bin`.

You can use one of the downloaded files, `gosales_scriptplayer.lst`, to run the action logs in sequence. This action generates a model named gosales_scriptplayer and publishes a package to the content store.

The action logs are described here.

01gosaddsrc.xml
Creates the model and adds a data source.

02goslangdef.xml
Defines the languages used by the model.

03gosmodqs.xml
Modifies a query subject.

04gosrenam.xml
Renames columns.

05gosprops.xml
Updates properties.

06gosorg.xml
Adds namespaces.

07goslangimp.xml
Imports a set of translations using text files stored in the same location.

08gospac.xml
Creates and publishes a gosales_scriptplayer package.

To run the script player, open a command prompt in *installation_location*/bin and run the following command:

```
BmtScriptPlayer -l gosales_scriptplayer.lst
```

Metadata Service

Framework Manager and IBM Cognos components communicate through the BI Bus API. A client issues requests and a service returns responses in the form of standard Simple Object Access Protocol (SOAP) messages. BI Bus API messages are XML documents encapsulated as SOAP requests that use the HTTP transport protocol.

To create your own BI Bus API messages, you must adhere to the Metadata Service Request schema and the Actions reference material.

When a client sends a BI Bus API request to the IBM Cognos Analytics server, the dispatcher routes the request to the Metadata Service. The Metadata Service is also responsible for encoding responses with SOAP before sending them back through the BI Bus API.

Using the BI Bus API messages, the Metadata Service can execute the actions that modify a model. The service can also query a model and return responses to your client. The Metadata Service responds with an XML document that contains the results of actions.

You can send two types of requests to the Metadata Service to manipulate an unpublished model:

- Send generic requests to create, open, save and close the model. Generic requests use the Framework Manager API request element.
- Send action requests to modify the metadata or publish a package. Action requests use the Framework Manager API request element.

To prepare these requests, you can use the Framework Manager API request element in your BI Bus API message with the Metadata Service.

Framework Manager API request element

You use the Framework Manager API request element in a BI Bus API element to notify the Metadata Service that the request contains a set of actions.

Here is an example of an Framework Manager API element showing an action request:

```
<mdprovider type=action model=GOSales.cpf action=execute>
  <transaction seq=1>
    <action seq=1 type=Delete>
      <inputparams>
        <param seq=1 type=handle>
          <value>[GoSales].[<Query Subject Name>]</value>
        </param>
      </inputparams>
    </action>
  </transaction>
</mdprovider>
```

Here is an example of an Framework Manager API element showing a generic request:

```
<mdprovider type=generic model=GOSales.cpf action=openModel></mdprovider>
```

Error handling

For each SOAP request, a response or fault is returned.

Here is an example of a successful SOAP request with returned parameters:

```
<response>
 <status success=true />
  <outputparams seq=1 type=handle value=[GoSales].[QuerySubjectName]
/>
```

```
   <outputparams seq=2 type=integer value=1 />
</response
```

Here is an example of an unsuccessful SOAP request:

```
<SOAP-ENV:Fault>
 <faultcode>Client</faultcode>
 <faultstring>BmtMDProviderRequestHandler caught BmtException</faultstring>
 <detail>
   <bus:exception>
   <severity>error</severity>
   <errorCode>MDPError</errorCode>
   <errorCodeString>Action: Delete failed, skipping... Reason:
Invalid
property handle ID:[GoSales].[QuerySubjectName]
   </errorCodeString>
   </bus:exception>
 </detail>
</SOAP-ENV:Fault>
```

Action logs

An action log is an XML document that contains a set of transactions. Each transaction contains one or more actions. Each action has a name and input parameters. Some actions also have output parameters.

For more information, see “Transactions” on page 7 and “Actions” on page 8.

You can use the Script Player or the Framework Manager application to play these action logs. You can choose to play back individual transactions or a combination of transactions in an action log.

When you use the Metadata Service, you send requests through the BI Bus API. These requests contain one or more actions in the same format as the actions in an action log. For example, you make changes to a project in a test environment. When it is time to move the project to production, you can play back every action, or series of actions, that you performed in the project in the test environment to create an identical project in the production environment.

In the Framework Manager application, the action log is stored in the project logs folder. The naming convention for the action log is the name of the project with the timestamp appended. For example, *<projectname>-<timestamp>*`-log.xml`.

For an example of a Framework Manager action log, see “Example - adding a security filter” on page 11.

Transactions

A transaction is a sequence of actions that is treated as a unit to satisfy a request. If any action fails, the entire transaction fails, and the actions already done in that transaction are rolled back.

A transaction is designated as a transaction by the transaction boundaries. These boundaries are unique to the method that uses the transactions.

For example, in the Framework Manager application you can create a folder and add query subjects into the folder. From your perspective, this is one request. From the Framework Manager perspective, this transaction is a series of actions grouped together. The action log shows these actions grouped together in one transaction.

In the Framework Manager API, transaction boundaries are determined differently by the Script Player, the Metadata Service, and the Framework Manager application.

In the Framework Manager application, a transaction sends a request, in the form of a set of actions, to the IBM Cognos Analytics server. The transaction is recorded in an action log (see “Action logs” on page 7) as an XML element. Each transaction element has a sequence number. The order of the transactions in the action log is significant. One example of how a series of actions is designated as a transaction in the Framework Manager application is the **Import** wizard. From the point at which the wizard is launched, until you click **OK**, a single transaction is created.

In the Script Player, transaction boundaries are explicit in the action log.

In the Metadata Service, a transaction boundary is a single SOAP request. One SOAP request is one transaction.

Actions

An action is a request made to the Framework Manager API. Actions are XML elements that contain input parameters. Some actions also have output parameters. Actions are defined in the `CR1Behaviors.xml` file, available in the *c10_location*`\templates\bmt\Cr1Model` directory. You can view some examples and actions documentation in the mdActions.xsd file, available in the *c10_location*`\templates\bmt\FMSDK` directory.

Actions are logged when you use the Framework Manager application. You can use these action logs with the Script Player. When you use the Metadata Service, you send requests through the BI Bus API. You can obtain requests from existing action logs (see "Action logs" on page 7).

By default, all actions that change the state of a Framework Manager metadata model are recorded in the log files. An example of these actions are DBImport, and Modify.

Some actions do not change the state of the model in the Framework Manager application and are not typically recorded in the action logs. An example of actions that are not recorded are DBBrowse and Publish. There are also some actions that are recorded but they do not change the state of the model. An example of this type of action is DBRelease.

Modifying the log status of actions

You can modify the log status of an action to determine whether or not you want it to appear in the action logs.

Procedure

1. Open the `CR1Behaviors.xml` file in the available in the *<c10_location>*`\templates\bmt\Cr1Model` directory.
2. Locate the `PluginList Version="0.2"` element. All actions are defined within this element.
3. Locate an action and check the value of the `loglevel` attribute.

   For example, the Publish action appears as `<Action Name="Publish" ID="10" loglevel="1"/>`

   A value of 1 means the action is not recorded in the action logs. A value of 2 means the action is recorded.
4. Modify the `loglevel` attribute as required.

   Framework Manager must be restarted for the change to take effect.

Objects you will use

When you work in Framework Manager, you work with a number of objects that are contained in a project.

Projects

A project contains a model, namespaces, packages, data sources, and related information for maintaining and sharing model information. A single project can span many data sources or tables.

An IBM Cognos Framework Manager project displays as a folder that contains a project file (.cpf) and the specific .xml files that define the project. The files in a project folder are unique to each project. The project and its associated files are contained in a project folder.

In general, do not add secondary files to the project folder because they may be affected by actions such as move, rename, and delete commands on the **Manage Projects** menu. If you decide to add secondary files to the project folders, the files are added with absolute paths. If they are moved from the original location, they must be retargeted.

These are the contents of a project folder.

***<project name>*.cpf**
The Framework Manager project file, which references the .xsd and .xml files that define a project.

archive-log.xml
This file contains the portion of the main log file that was archived.

customdata.xml
This file contains the layout information for the diagram.

If this file is deleted, layout information is lost. An automatic layout will be applied.

IDLog.xml
This file tracks objects for models that use branching and merging.

log.xml
A list of all modifications made to the model.

mda_metadata.xml
A Model Design Accelerator file, which contains the metadata imported from data sources.

mda_engine_project.xml
A Model Design Accelerator file, which contains the definition of the star schema.

model.xml
The actual model data created by Framework Manager users.

preferences.xml
The preferences for Framework Manager projects.

session-log.xml

A list of unsaved transactions in the model. When the project is saved, this list is deleted. View contents of this file using View Transaction History.

When Framework Manager is started, the existing session-log.xml file is renamed to `session-log-backup.xml`.

session-log-backup.xml

The `session-log.xml` from the previous session. Using this file, a modeler can run a script to restore the unsaved model transactions in the event of an unexpected interruption in the current session.

This file is deleted each time Framework Manager is started. Ensure you make a copy of this file before exiting the current Framework Manager session if you want to keep a copy.

repository.xml
The logged version history for each project or segment that was added to a repository; this file exists only if you added projects to a repository.

upgradeReport.htm
The content of the upgrade summary message that is displayed after upgrade.

Models

A model is the set of related dimensions, query subjects, and other objects required for one or more related reporting applications.

The Framework Manager model is a metadata layer that adds value to a data source in several ways. Most importantly, it provides a business view of the information in the source data to simplify building reports, analyses, and queries. The business view can:

- Organize items in folders that represent business areas for reporting
- Format items using numeric, currency, date, time, and other formats
- Present multilingual folder and item names, descriptions, tips, and data so that users can operate in their language of choice

- Automate the generation of SQL queries sent to the relational data source
- Specify default prompting

  This can include having IBM Cognos software prompt the user using a descriptive name while actually filtering on a code or key value for improved query performance.

In particular, you can modify the Framework Manager model to ensure that queries sent to the data source are efficient, well formed, and secure. You can specify the rules governing query generation, restrict user access to specific rows or columns of data, and model data relationships to hide the complexity of data from your users.

Namespaces

A namespace uniquely identifies query items, dimensions, query subjects, and other objects. You import different databases into separate namespaces to avoid duplicate names.

Folders

A folder is a grouping of metadata objects that, unlike namespaces, does not affect the identification of its contained objects. For example the identifier used for a query subject does not change if the object is moved into or out of a folder.

Packages

A package is a subset of the dimensions, query subjects, and other objects defined in the project. A package is what is actually published to the IBM Cognos Analytics server, and it is used to create reports, analyses, and ad hoc queries.

Dimensions

A dimension is a broad grouping of data about a major aspect of a business, such as products, dates, or markets.

The types of dimensions that you can work with in IBM Cognos Framework Manager are regular dimensions and measure dimensions. In SAP BW, measure dimensions are called key figures.

Query subjects

A query subject is a set of query items that have an inherent relationship.

In most cases, query subjects behave like tables. Query subjects produce the same set of rows regardless of which columns were queried.

There are different types of query subjects.

Data source
: Data source query subjects directly reference data in a single data source. IBM Cognos Framework Manager automatically creates a relational data source query subject for each table and view that you import into your model.

Model
: Model query subjects are not generated directly from a data source but are based on query items in other query subjects or dimensions, including other model query subjects. By using model query subjects, you can create a more abstract, business-oriented view of a data source.

Stored procedure
: Stored procedure query subjects are generated when you import a procedure from a relational data source. IBM Cognos Framework Manager supports only user-defined stored procedures. System stored procedures are not supported.

Query items

A query item is the smallest piece of the model that can be placed in a report. It represents a single characteristic of something, such as the date that a product was introduced.

Query items are contained in query subjects or dimensions. For example, a query subject that references an entire table contains query items that represent each column in the table.

For your users, query items are the most important objects for creating reports. They use query item properties of query items to build their reports.

Governors

Framework Manager governors are used to reduce system resource requirements and improve performance. The governor settings are specified before packages are created to ensure that the metadata in the package contains the specified limits. If governor settings are not specified, the default values are used.

For more information, see "Governors" in the *IBM Cognos Framework Manager Guide*.

Example - adding a security filter

To learn more about the Framework Manager API, we recommend that you be familiar with the Framework Manager application. Use the Framework Manager application to perform the modeling tasks you need, and review the log file to see the results.

After you understand how actions are used, you can create your own action logs to accomplish similar goals. The last code sample in this example demonstrates how you can combine some actions that the Framework Manager application needs to separate.

Using the Framework Manager application

In this Framework Manager action log example, you apply a security filter to a query subject in the model.

In this action log example, there is one transaction. The transaction contains three actions. Two of the actions are partly duplicated because the Framework Manager application executes two ModifyComplex actions. One action identifies the user, the other action identifies the object.

The AddProperty and the first ModifyComplex actions add a group or user to a query subject. The second ModifyComplex action adds the actual security filter.

Transaction

This sample code shows the structure of the transaction. This transaction contains three actions, as required by the Framework Manager application: AddProperty, ModifyComplex, and ModifyComplex.

```
<transaction seq=1 timestamp=20030512093334778>
  <action seq=1 type=AddProperty>
    ...
  </action>
  <action seq=2 type=ModifyComplex>
    ...
  </action>
  <action seq=3 type=ModifyComplex>
    ...
  </action>
</transaction>
```

First Action - AddProperty

This code sample shows that the securityFilters property (querySubject/securityFilters) is added to the querySubject object ([oracle_gosales].[COUNTRY]):

```
<action seq=1 type=AddProperty>
<inputparams>
  <param seq=1 type=handle>
    <value>[oracle_gosales].[COUNTRY]</value>
  </param>
  <param seq=2 type=i18nstring>
    <value>querySubject/securityFilters</value>
  </param>
</inputparams>
 <domchanges/>
<result success=t>
  <outputparams/>
</result>
</action>
```

Second Action - ModifyComplex

This code sample shows that the new securityFilters property is modified to include a securityFilterDefinition. The /0/ is a text separation sequence used by the parser to recognize parts of the value element.

```
<action seq=2 type=ModifyComplex>
<inputparams>
<param seq=1 type=handle>
  <value>/0/securityFilters[0]/0/[oracle_gosales].[COUNTRY]</value>
</param>
<param seq=2 type=i18nstring>
<value>
&lt;securityFilterDefinition&gt;
  &lt;securityObject type=&quot;account&quot;&gt;
    &lt;displayPath&gt;
      firstName lastName(userID)[Directory &amp;gt; LDAP &amp;gt;People]
    &lt;/displayPath&gt;
    &lt;cmSearchPath&gt;
      CAMID(&amp;quot;LDAP:u:uid=userID,ou=people&amp;quot;)
    &lt;/cmSearchPath&gt;
  &lt;/securityObject&gt;
  &lt;expression/&gt;
&lt;/securityFilterDefinition&gt;
  </value>
</param>
</inputparams>
  <domchanges/>
<result success=t>
  <outputparams/>
</result></action>
```

The contents of <param seq=2 type=i18nstring><value> are encoded. The translation of the encoding is

```
<securityFilterDefinition>
<securityObject type=''account''>
  <displayPath>firstName lastName(userID)[Directory > LDAP >
People]
  </displayPath>
  <cmSearchPath>CAMID(''LDAP:u:uid=userID,ou=people'')
  </cmSearchPath>
</securityObject>
<expression/>
</securityFilterDefinition>
```

Third Action - ModifyComplex

This code sample shows that the actual security filter is added to the securityFilterDefinition:

```
<action seq=3 type=ModifyComplex>
<inputparams>
<param seq=1 type=handle>
```

```
  <value>/O/securityFilters[0]/O/[oracle_gosales].[COUNTRY]</value>
</param>
<param seq=2 type=i18nstring><value>
&lt;securityFilterDefinition&gt;
    &lt;securityObjecttype=&quot;account&quot;&gt;
      &lt;displayPath&gt;
       firstNamelastName(userID)[Directory&amp;gt; LDAP &amp;gt;People]
      &lt;/displayPath&gt;
      &lt;cmSearchPath&gt;
        CAMID(&amp;quot;LDAP:u:uid=userID,ou=people&amp;quot;)
      &lt;/cmSearchPath&gt;
   &lt;/securityObject&gt;
   &lt;expression&gt;
     &lt;refobj&gt;
      [oracle_gosales].[COUNTRY].[COUNTRY]
     &lt;/refobj&gt;
     like&amp;apos;Canada&amp;apos;
   &lt;/expression&gt;
&lt;/securityFilterDefinition&gt;
</value>
</param>
</inputparams><domchanges/>
  <result success=t>
<outputparams/>
</result>
</action>
```

The contents of <param seq=2 type=i18nstring><value> are encoded. The translation of the encoding is

```
<securityFilterDefinition>
<securityObject type=''account''>
  <displayPath>firstName lastName(userID) [Directory > LDAP
> People]</displayPath>
  <cmSearchPath>CAMID(''LDAP:u:uid=userID,ou=people'')
  </cmSearchPath>
</securityObject>
<expression><refobj>[oracle_gosales].[COUNTRY].[COUNTRY]</refobj>like
'Canada' </expression>
</securityFilterDefinition>
```

Example - complete action log

This code sample shows the entire action log. You can reuse this action log in the Framework Manager application, as well as by the Script Player and the Metadata Service. To use this file with the Metadata Service, you must encode the action log in a SOAP envelope.

For more information, see Chapter 4, “Metadata Provider (Wrapper) reference,” on page 115 .

```
<bmtactionlog user=userID timestamp=20030512094158162>
  <transaction seq=1 timestamp=20030512093334778>
   <action seq=1 type=AddProperty>
    <inputparams>
<param seq=1 type=handle>
<value>[oracle_gosales].[COUNTRY]</value>
</param>
<param seq=2type=i18nstring>
<value>querySubject/securityFilters</value>
</param>
</inputparams>
<domchanges/>
<result success=t>
<outputparams/>
</result>
</action>
<action seq=2 type=ModifyComplex>
<inputparams>
<param seq=1 type=handle>
<value>/O/securityFilters[0]/O/[oracle_gosales].[COUNTRY]</value>
</param>
<param seq=2type=i18nstring><value>&lt;securityFilterDefinition&gt;
&lt;securityObject
type=&quot;account&quot;&gt;&lt;displayPath&gt;firstName
lastName(userID)
[Directory &amp;gt; LDAP &amp;gt;
People]&lt;/displayPath&gt;&lt;cmSearchPath&gt;CAMID(&amp;quot;LDAP:u:uid=user
ID,ou=people&amp;quot;)&lt;/cmSearchPath&gt;&lt;/securityObject&gt;&lt;express
```

```
ion/&gt;&lt;/securityFilterDefinition&gt;</value></param>
</inputparams>
<domchanges/>
<result success=t>
<outputparams/>
</result>
</action>
<action seq=3 type=ModifyComplex>
<inputparams>
<param seq=1type=handle>
<value>/O/securityFilters[0]/O/[oracle_gosales].[COUNTRY]</value>
</param>
<param seq=2type=i18nstring><value>&lt;securityFilterDefinition&gt;
&lt;securityObject
type=&quot;account&quot;&gt;&lt;displayPath&gt;firstName
lastName(userID)
[Directory &amp;gt; LDAP &amp;gt;
People]&lt;/displayPath&gt;&lt;cmSearchPath&gt;CAMID(&amp;quot;LDAP:u:uid=user
ID,ou=people&amp;quot;)&lt;/cmSearchPath&gt;&lt;/securityObject&gt;&lt;express
ion&gt;&lt;refobj&gt;[oracle_gosales].[COUNTRY].[COUNTRY]&lt;/refobj&gt;
like
&amp;apos;Canada&amp;apos;&lt;/expression&gt;&lt;/securityFilterDefinition&gt;
</value></param>
</inputparams>
<domchanges/>
<result success=t>
<outputparams/>
</result>
</action>
</transaction>
</bmtactionlog>
```

Example - creating a simplified action log

The user interface needs two ModifyComplex actions to accomplish this task. However, if the actions are executed programmatically, the first ModifyComplex is not necessary. One ModifyComplex is sufficient to identify the user and the object.

```
<bmtactionlog user=userID
timestamp=20030512094158162>
<transaction seq=1 timestamp=20030512093334778>
<action seq=1 type=AddProperty>
<inputparams>
<param seq=1 type=handle>
   <value>[oracle_gosales].[COUNTRY]</value>
</param>
<param seq=2 type=i18nstring>
  <value>querySubject/securityFilters</value>
</param>
</inputparams>
<domchanges/>
<result success=t>
<outputparams/>
</result>
</action>
<action seq=2 type=ModifyComplex>
<inputparams>
<param seq=1 type=handle>
   <value>/O/securityFilters[0]/O/[oracle_gosales].[COUNTRY]</value>
</param>
<param seq=2 type=i18nstring>
  <value>&lt;securityFilterDefinition&gt;&lt;securityObject
type=&quot;account&quot;&gt;&lt;displayPath&gt;firstName
lastName(userID)
[Directory &amp;gt; LDAP &amp;gt;
People]&lt;/displayPath&gt;&lt;cmSearchPath&gt;CAMID(&amp;quot;LDAP:u:uid=user
ID,ou=people&amp;quot;)&lt;/cmSearchPath&gt;&lt;/securityObject&gt;&lt;express
ion&gt;&lt;refobj&gt;[oracle_gosales].[COUNTRY].[COUNTRY]&lt;/refobj&gt;
like
&amp;apos;Canada&amp;apos;&lt;/expression&gt;&lt;/securityFilterDefinition&gt;
</value></param>
</inputparams>
<domchanges/>
<result success=t>
<outputparams/>
</result>
</action>
```

```
</transaction>
</bmtactionlog>
```

Chapter 2. Creating custom report functions and function sets

This chapter describes how to create custom report functions and custom function sets for report authors to use in IBM Cognos Analytics.

Creating custom report functions

Report authors create report expressions using the expression editor. The expression editor provides a list of functions that can be used in expressions. In addition to the functions that are available by default, such as Today(), ReportDate() or ReportName(), you can create custom functions and make them available to report authors by defining report function libraries.

All functions available through the expression editor must be identified in the function definition service, a series of files that tells the expression editor what functions are defined. Report function collections are provided to report authors through dynamic load libraries on Windows operating systems, in shareable libraries on UNIX operating system, or in shared object files on the Linux operating system.

Many types of function metadata definitions are shared among IBM Cognos applications and their components. Only the requirements for defining custom report functions are outlined in this chapter.

To make custom report functions available to report authors, you:

- Build a custom report functions library "Building a custom report functions library" on page 17
- Register the report functions library "Registering custom report functions" on page 20
- Install the custom report functions "Installing a custom report functions library" on page 22

For an example, see "Example of a custom report functions implementation" on page 23.

Building a custom report functions library

Custom report functions can be built in any programming language that allows creation of the appropriate file type - dynamic load libraries on Windows operating systems, shareable libraries on UNIX operating system, or shared object files on the Linux operating system.

The report function declaration must follow a specific format, as defined in the `crxSDK.h` file. In compiling your .dll files, this header file is always included via an include statement.

Example of a report function prototype

Report functions may have any number of arguments, ranging from none to 15. The report function prototype, consisting of its name and arguments, is defined as follows in the `crxSDK.h` file.

```
typedef CCLDBColumnState (*PF_CallFunction)
(
void* result,
uint resultsize,
const crxDataI* context,
void* arg1,
void* arg2,
```

```
void* arg3,
void* arg4,
void* arg5,
void* arg6,
void* arg7,
void* arg8,
void* arg9,
void* arg10,
void* arg11,
void* arg12,
void* arg13,
void* arg14,
void* arg15,
void* arg16
);
```

where:

- The first argument receives the result of the function execution.
- The second argument, resultsize, sets the size of the results buffer in bytes. The buffer is pre-allocated by the expression engine.
- The third argument is the context, and is ignored.
- All other arguments are the function input arguments as specified in the function definition file. "Function definition file" on page 20.

Result and function arguments

The result and the function input arguments are pointers to any of the types shown in this table.

Table 2. Result and Function Argument types

Supported Types	**Comments**
CCL_int8	
CCL_uint8	
CCL_int16	
CCL_uint16	
CCL_int32	

Table 2. Result and Function Argument types (continued)	
Supported Types	**Comments**
CCL_uint32	
CCL_int64	
CCL_uint64	
CCL_float32	
CCL_float64	
CCL_char[CRX_MAXIMUM_STR_CODEPOINTS]	for strings
CCL_uint8[CRX_MAX_DECIMAL_BYTES]	for decimals
CCLDate2	
CCLTime2	
CCLTimeTZ	
CCLDateTime	
CCLTimeStamp2	
CCLTimeStampTZ	
CCLIntervalYM	
CCLInterval2	

Note: For the definition of these types, see the `crxSDK.h` file located in the *`installation_location`*`/webcontent/samples/sdk/crx/crxSDKsample` directory.

Context argument

The `context` argument points to a helper object that the expression engine and its client application, IBM Cognos Analytics, use to handle the variables that can be specified in report expressions. It is used:

- At compilation time, to resolve variables by name and retrieve their properties (type, size, precision, scale).
- At execution time, to retrieve the variables' values.

Custom functions do not use the `context` argument. For these functions, this argument is always null.

Report function return value

The value returned by the report function call, of type `CCLDBColumnState`, tells the expression engine the status of the function execution. `CCLDBColumnState` may take any of the following values.

CCL_DB_COLSTATE_OK
Function call was successful.

CCL_DB_COLSTATE_NULL
One of the function arguments was missing (NULL).

CCL_DB_COLSTATE_NA
One of the function arguments was unavailable.

CCL_DB_COLSTATE_DIVBYZERO
A divide-by-zero error occurred.

CCL_DB_COLSTATE_OVERFLOW
For numerics, an overflow or underflow occurred. For strings, truncation of the string occurred.

CCL_DB_COLSTATE_SECURITY
Access to one of the function arguments was prohibited for security reasons.

CCL_DB_COLSTATE_UNKNOWN
Status is reserved for cases where the status is not truly known.

CCL_DB_COLSTATE_ERROR
A generic error indicating all other cases.

CCL_DB_CASTING_ERROR
Invalid data was passed to a data type casting function.

CCL_DB_COLSTATE_SAMPLE
Temporary status returned by the engine while processing is not yet complete. Internal use only.

Registering custom report functions

After building the dynamic load libraries, you must register the functions so the expression engine can recognize the custom functions.

To register the functions, you must perform the following tasks:

- Create a custom file list "File list" on page 20.
- Create a function definition file "Function definition file" on page 20.
- Create one or more function description files "Function description files" on page 22.

The default files used by the Function Definition Service are located in the *installation_location*/configuration/functions directory. You can use these as models to create your custom files.

For an example, see "Example of a custom report functions implementation" on page 23.

To register your custom functions after an upgrade, the custom function files can simply be copied back into the functions directory.

As in previous releases, you can still add custom functions to a default group. However, changes to the Function Definition Service are not retained after you upgrade to another version of IBM Cognos Analytics. If you modify a default group and then upgrade IBM Cognos Analytics, you will have to recreate your custom functions.

File list

FileList.xml is the default file where all function definition files provided by IBM Cognos are listed. For custom functions, you must create a similar file with the name pattern of filelist*n*.xml, where *n* is any name that you assign. For example,

```
filelist_custom.xml
```

This file will contain a list of your custom function definition files and related function description files.

Function definition file

For each entry in your custom filelist, you must create a function definition file.

Every report function and its function element must be unique in the entire series of function definition files. To avoid duplication of functionality, check that the functions you require don't already exist in the default definition files. The default files are cogRSReportFunctions.xml and cogCRXReportFunctions.xml.

The schema file for the function definition file is FuncTree.xsd. It is located in the *installation_location*/configuration/functions directory.

Every group of functions is described by a group element uniquely identified by the value of its id element. Every report function in the group must be described within a function element. The function element contains these elements described here.

id
A unique string identifier designated by the developer and used internally by the expression engine. The function description files also use this identifier. The id element must be unique across the entire set of IBM Cognos functions.

name
The name that shows up in the expression editor tree controls. It can be overridden by an entry in the language file.

canonical
The name of the report function as defined in the dynamic load library.

dll
The library name. The extension (.dll, .so) is not included if it matches the default for the platform. The default path for the library is the bin directory. You can also specify a path relative to the default path.

context
This element is used internally by the definition service, and must always be set to CRX.

returnType
The type for the value returned by the function to the user. It must map to the result argument type as specified in the report function declaration. For more information, see "Type mapping" on page 21.

parameter
Optional. Used to describe function input arguments. Every parameter must contain a type element.

type
Sub-element of the parameter element. Must map to the report function argument type. For more information, see "Type mapping" on page 21.

Type mapping

There is a direct one to one mapping between the returnType and type elements in the function definition file and the argument type as defined in the .dll file.

Possible types for returnType and type elements and their corresponding mappings are shown in the following table.

Table 3. returnType and type mappings

returnType or type element	**Report function argument type from crxSDK.h file**
crxDTypeInt8	CCL_int8
crxDTypeUInt8	CCL_uint8
crxDTypeInt16	CCL_int16
crxDTypeUInt16	CCL_uint16
crxDTypeInt32	CCL_int32
crxDTypeUInt32	CCL_uint32
crxDTypeInt64	CCL_int64
crxDTypeUInt64	CCL_uint64
crxDTypeFloat	CCL_float32

Table 3. returnType and type mappings (continued)	
returnType or type element	**Report function argument type from crxSDK.h file**
crxDTypeDouble	CCL_float64
crxDTypeString	CCL_char[CRX_MAXIMUM_STR_CODEPOINTS]
crxDTypeDecimal	CCL_uint8[CRX_MAX_DECIMAL_BYTES]
crxDTypeDate	CCLDate2
crxDTypeTime	CCLTime2
crxDTypeTimeTZ	CCLTimeTZ
crxDTypeDatetime	CCLTimeStamp2
crxDDatetimeTZ	CCLTimeStampTZ
crxDTypeYMInterval	CCLIntervalYM
crxDTypeDTInterval	CCLInterval2

For the definition of these types, see the crxSDK.h file located in the *installation_location*/webcontent/samples/sdk/crx/crxSDKsample directory.

Function description files

For each function definition file, you need to create at least one function description file. There must be one description file for each supported language. Each file contains the function name, syntax, and tip for a particular language. The functions described in these files are cross-referenced by the id attribute of the function element.

Each function description file is named by combining the file name and a locale identifier, separated by an underscore. If only one description file is provided, the locale identifier must be en. The contents of the file, however, can be in any language.

For example, if the function definition file name is crxSDKSampleTree.xml, then the function description files could be named crxSDKSampleStrings_*xx*.xml where *xx* stands for any locale identifier, such as en for English or ja for Japanese.

The i18n_res.xml file, located in the bin directory, contains the list of locales and their identifiers. If this file is missing, IBM Cognos Analytics substitutes a standard list of default locales: en (English), fr (French), de (German), and ja (Japanese).

The English function description file is the default. If the requested localized file does not exist, the English file will be used.

The content of the function description files is used in the expression editor. Each function description has three parts that are described here.

function name
Identifies the function in the functions list in the expression editor.

syntax
Describes the exact format and required parameters that must be entered by the report author.

tip
Describes what the function does.

Installing a custom report functions library

IBM Cognos Analytics recognizes your custom functions automatically, once you ensure that the three function files you create are located in the *installation_location*/configuration/functions directory.

To install your custom functions after an upgrade, the custom function files can simply be copied back into the functions directory.

The location that you specify in the `dll` element of a function definition file tells the expression engine where to find the corresponding dynamic load library or libraries. You must ensure that this specification matches the location of the .dll files. If you specify the file name only, ensure that the .dll files are located in the *`installation_location`*`/bin` directory.

Example of a custom report functions implementation

This topic illustrates the implementation of a set of custom report functions.

The Sample files are located in the *`installation_location`*`/webcontent/samples/sdk/crx/crxSDKsample` directory. They are described in the following table.

Table 4. Custom report function sample files

File Name	Purpose	Description
`crxSDKSample.cpp`	sample C++ code	Is required to add the custom report functions. It will be compiled into a dynamic load library and referenced in the function definition file
`crxSDK.h`	header file	Contains the report function prototype, and the type definitions used by CRX report functions. It must be referenced by an include statement in each dynamic load library you create for custom report functions.
`FileList_custom.xml`	custom file list file	Identifies the custom function definition and description files.
`crxSDKSampleTree.xml`	function definition file	Represents a custom group of functions. The file identifies the functions and their location to FDS.
`crxSDKSampleStrings_en.xml`	function description file	Provides the strings that will appear to the report author in the expression editor. In this example, only an English description file is included.

The functions are:

- A random integer generator
- A surface area calculator
- A date to a string converter

The following table shows the report function declaration and the function name and syntax that the report author sees in the expression editor.

Table 5. Report function declarations

Function Declaration in C++	Function Name and Syntax
`CCLDBColumnState SDKRandomInt ( void* result, const crxDataI* context);`	`SDKRANDOMINT()`

Table 5. Report function declarations (continued)	
Function Declaration in C++	**Function Name and Syntax**
`CCLDBColumnState SDKArea ( void* result, const crxDataI* context, const CCL_float64 arg1 const CCL_float64 arg2 const CCL_char arg3 );`	`SDKAREA ([base],[height],[shape])`
`CCLDBColumnState SDKDateToString ( void* result, const crxDataI* context const QSDate* date`	`SDKDATETOSTRING(date)`

Example of a custom file list file

The following is a segment of the custom file list.

```
<filelist xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
                xsi:noNamespaceSchemaLocation="FileList.xsd">
                <definition>crxSDKSampleTree.xml</definition>
    <description>
        <language>en-us</language>
        <file>crxSDKSampleStrings_en.xml</file>
    </description>
</filelist>
```

Example: crxSDKSampleTree.xml

The following is a segment of the function definition file showing the definition of the `Area` function.

```
...
<functionsRoot xsi:schemaLocation=
"http://www.developer.cognos.com/schemas/commonfunctionservice/1/0 FuncTree.xsd"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
<group>
<id>SDKTEST</id>
<!--CRX SDK Test Functions-->
...
```

```
<function>
   <id>crx_sdk_area</id>
   <canonical>SDKArea</canonical>
   <name>SDKArea</name>
   <dll>CRXSampleSDKFunctions</dll>
   <context>CRX</context>
   <returnType>crxDTypeDouble</returnType>
     <parameter>
      <type>crxDTypeDouble</type>
     </parameter>
     <parameter>
      <type>crxDTypeDouble</type>
     </parameter>
     <parameter>
      <type>crxDTypeString</type>
</function>
...
</group>
</functionsRoot>
```

Example: crxSDKSampleStrings_en.xml

The following is a segment of the English function description file showing the Area function.

```
...
<stringTable xsi:noNamespaceSchemaLocation="CCLMessageFile.xsd"
usage="String"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
<component name="CRN">
```

```
<section name="TST">
...
```

```
<string id="crx_sdk_area">SDKAREA</string>
<string id="crx_sdk_area.syntax">SDKAREA(numeric_exp, numeric_exp, shape)</string>
<string id="crx_sdk_area.tip">Returns value of the area of a
Parallelepiped (P) or a Triangle (T).</string>
...
</section>
</component>
</stringTable>
```

Creating custom report functions

You can create custom report functions with the following steps.

Procedure

1. Define the functions (`crxSample.cpp`). The header file (`crxSDK.h`) is included in the program.
2. Compile the program into a dll, shared library or shared object file (`CRXSampleSDKFunctions.dll`).
3. Create a custom filelist (`filelist_custom.xml`).
4. Create a new function definition file (`crxSDKSampleTree.xml`).
5. Create a new function description file (`crxSDKSampleStrings_en.xml`).

Custom function sets

A function set is a collection of database functions that are vendor-specific. You can customize the Function Description Service to expose or define functions that your data source supports. Defining additional vendor-specific functions is similar to defining custom functions. The following elements are used to specify the vendor group.

id
For a custom function set, the value of the id is `vendor_group`. Groups with the same identifier in the same position of the tree are merged.

context
Defining a context restricts the visibility of this group when the application specifies a particular context.

vendor
Identifier for the vendor. This restricts the visibility of this group when the application specifies a particular vendor.

datasourceQueryType
Underlying database query technology.

Creating a custom function set

You can create a custom function set with the following steps.

Procedure

1. Create a custom filelist named `filelist_customerExtensions.xml` for the vendor extensions.
2. Create a functions definition file named `customerExtensionsTree.xml` in the *`installation_location`*`/configuration/functions` directory. The contents of `customerExtensionsTree.xml` are validated against the `FuncTree.xsd` file.
3. Create a function description file named `customerExtensionsString_en.xml`.

Example of creating a custom function set

The following example shows how to add the `datepart` SQL Server function to the list of functions displayed in the expression editor. This function is not delivered as part of the default FDS content because there is an equivalent extract function in the SQL99 folder.

Create a filelist_customerExtensions.xml file

For adding custom function sets, a custom filelist is created the same way as for custom functions. A single custom filelist could be used for all your custom function definition and description files.

```
...
<filelist xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:noNamespaceSchemaLocation="FileList.xsd">
    <definition>customerExtensionsTree.xml</definition>
    <description>
        <language>en-us</language>
        <file>customerExtensionsString_en.xml</file>
    </description>
</filelist>
```

Create a CustomerExtensionsTree.xml file

Create a `customerExtensionsTree.xml` file that contains the following content. Note that the file structure mimics the `SQLServerTree.xml` file, so that the content of folders can come from multiple files.

```
<?xml version="1.0" encoding="UTF-8"?>
<functionsRoot xsi:schemaLocation=
"http://www.developer.cognos.com/schemas/commonfunctionservice/1/0FuncTree.xsd"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
<group>
<id>vendor_group</id>
<group><id>SQLServer</id>
    <context>tabular</context>
    <vendor>V_SQLServer</vendor>
    <datasourceQueryType>relational</datasourceQueryType>
<group>
```

```
<id>ext_sql</id>
<!--SQL Server Extensions. There are probably equivalent SQL99
functions available for these-->
<function>
   <id>ext_datepart</id>
   <canonical>datepart</canonical>
   <name>datepart</name>
     <returnType>numeric</returnType>
     <parameter>
     <type>string</type>
     </parameter>
     <parameter>
       <type>dateTime</type>
      </parameter>
</function>
</group>
</group>
</group>
</functionsRoot>
```

Create a CustomerExtensionsString_en.xml file

Create a `customerExtensionsString_en.xml` file that contains the following content. You can create additional description files for each language you want to support.

```
<?xml version="1.0" encoding="UTF-8"?>
<stringTable xsi:noNamespaceSchemaLocation="CCLMessageFile.xsd"
usage="String"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
```

```
<component name="EXT">
<section name="EXT" type="UI">
<string id="ext_sql">Additional SQL Server Functions</string>
     <string id="ext_datepart">datepart</string>
     <string id="ext_datepart.syntax">
       datepart ( { year | month | day }, datetime_exp )
     </string>
     <string id="ext_datepart.tip">
      Returns the numeric value of part of a date.
      Note the { } braces around the first parameter.
     </string>
</section>
</component>
</stringTable>
```

Chapter 3. Model schema reference

This section contains information about the elements and attributes that are used in the `model.xml` file, the xml representation of the model.

The `BMTModelSpecification.xsd` file, available in the *installation_location*`\templates\bmt\Cr1Model` directory, validates the `model.xml` file.

For each element, the following information is provided:

- A description of the element
- Descriptions of required and optional attributes
- The child elements that the element can or must have
- The parent elements that can contain the element

If you do not specify the value for an optional attribute that has a default value, Framework Manager automatically applies the attribute and uses the default value. We recommend that you review the attributes for the elements you want to use, and set the attribute values if you do not want to use the defaults.

For elements that contain multiple attributes, you may specify the attributes in any order within the element.

access

Defines who has access to a subset in the published model.

Content model

decisionRole *(optional)*

Parent elements

securityView

adminAccess

The set of security objects to which administrative permissions are granted on the published model, as defined in Content Manager. The permissions are Read, Write, Traverse, and setPolicy.

The original publisher of the package, its owner, is always granted these permissions.

Content model

decisionRole *(optional)*

Parent elements

package

aggregateRule

The aggregation rule that applies to this measure for the specified dimension.

Content model

dimensionRef then applyAggregate

Parent elements

aggregateRules

aggregateRules

A collection of aggregate rules.

Content model

aggregateRule *(any number)*

Parent elements

measure

aggregationRule

The additive aggregation rule. Indicates how the query item is aggregated in the data source. Unless otherwise specified, zero and NULL values are included in calculating the result.

Content model

Content type is NMTOKEN.

The possible values of this element are restricted to the following.

unsupported

Aggregation is not supported for this object.

unknown

Specifies that the aggregation type is unknown.

average

Aggregation is computed as an average of the values in the result set for this object.

automatic

The aggregation rule is determined automatically at run time. Wherever possible, the value of calculated is used, then sum where necessary, followed by unsupported.

calculated

Aggregated values are computed by aggregating the elements of the expression rather than the result. For example, if the aggregate rule for QI_a is set to sum, and the rule for QI_b is set to average, then if this query item is based on the expression (QI_a / QI_b) * 100, the aggregate is computed as (sum(QI_a) / average(QI_b)) * 100.

count

Aggregation is computed as a count of the items in the result set for this object.

countDistinct

Aggregation is computed as a count of unique items in the result set for this object.

countNonZero

Excludes zero and NULL values.

maximum

Aggregation determines the maximum value of the result set for this object.

median

Aggregation determines the median value of the result set for this object.

minimum

Aggregation determines the minimum value of the result set for this object.

standardDeviation

Aggregation determines the standard deviation for the result set for this object.

sum

Aggregation determines the sum of the values in the result set for this object.

variance

Aggregation determines the variance of the result set for this object.

Parent elements

calculation , measure , procParameter , procParameter , queryItem , result

aliasTableMapRef

References a `parameterMap` that represents a Hyperion Essbase alias table map.

Content model

Content type is string.

Parent elements

dataSource , dimension

allocation

Reserved.

Content model

Content type is NMTOKEN.

The possible values of this element are restricted to the following.

doNotAllocate

Reserved.

constant

Reserved.

Parent elements

measureScope

allocationRule

Specifies the type of allocation defined for the measure.

Content model

Content type is NMTOKEN.

The possible values of this element are restricted to the following.

default

Specifies that constant allocation is used in list queries and once-only allocation is used in crosstab queries.

constant

Specifies that constant allocation is used in all queries.

Parent elements

measure

applyAggregate

Aggregation to apply.

Content model

Content type is token.

The possible values of this element are restricted to the following.

unsupported

Aggregation is not supported for this object.

unknown

Specifies that the type is not known.

sum

Aggregation determines the sum of the values in the result set for this object.

minimum

Aggregation determines the minimum value of the result set for this object.

maximum

Aggregation determines the maximum value of the result set for this object.

average

Aggregation is computed as an average of the values in the result set for this object.

count

Aggregation is computed as a count of the items in the result set for this object.

firstPeriod

Returns the value associated with the first, lowest-level member in the current time period.

lastPeriod

Returns the value associated with the last, lowest-level member in the current time period.

standardDeviation

Aggregation determines the standard deviation for the result set for this object.

variance

Aggregation determines the variance of the result set for this object.

Parent elements

aggregateRule

attributeDimensionsAsProperties

If set to `true`, Hyperion Essbase attribute dimensions are to be presented as properties of other dimensions. If set to `false`, Hyperion Essbase attribute dimensions are to be presented as separate dimensions.

Content model

Content type is boolean.

Parent elements

dataSource

attributes

A collection of query items that are uniquely identified by the key.

Content model

refobj *(any number)*

Parent elements

determinant

balanced

A balanced hierarchy contains no skipped or empty levels. All branches of the dimension hierarchy tree descend through all existing levels, and each level has at least one member. All members at a given level have the same number of ancestors.

Content model

Content type is boolean.

Parent elements

hierarchy

basedOn

A filter, which at execution time must be logically compared with the filter in the object that has the matching `securityObject` using the AND operator.

Content model

cmSearchPath

Parent elements

securityFilterDefinition

calcType

If the calculation is imported from OLAP sources, it can represent either a named set or a substitution variable.

Content model

Content type is NMTOKEN.

The possible values of this element are restricted to the following.

unknown

Specifies that the calculation type is unknown.

namedSet

Specifies that the calculation represents a named set.

macros

Specifies that the calculation represents a substitution variable.

Parent elements

calculation

calculation

Defines an expression that returns a value.

The `externalName`, `calcType`, `hierarchies`, `dimensions` or `datasources` elements are only set if the calculation has been imported from an OLAP data source. If any of the these elements are set, the calculation is read-only.

Attributes

status

Specifies the status of the resolution of a calculation.

Usage: `optional` Default: `valid` Type: `NMTOKEN`.

Attribute values are restricted to the following.

valid

Specifies that the syntax of the object is valid.

invalid

Specifies that the syntax of the object is not valid.

needsReevaluation

Specifies that the object syntax needs to be reviewed.

Content model

(name *(one or more)* then description *(any number)* then (guid *(optional)* then lastChanged *(optional)* then lastChangedBy *(optional)* then comment *(optional)*) then screenTip *(any number)* then property *(any number)*) then (expression then (previewFilters *(optional)* then securityFilters *(optional)*) then (hidden *(optional)* then usage *(optional)* then format *(optional)* then currency *(optional)* then (datatype *(optional)* then precision *(optional)* then scale *(optional)* then size *(optional)* then nullable *(optional)* then aggregationRule *(optional)*) then displayType *(optional)* then MIMEType *(optional)* then promptInfo *(optional)* then regularAggregate *(optional)* then semiAggregate *(optional)* then sortOnRef *(optional)* then unSortable *(optional)* then roles *(optional)* then conformanceRef *(optional)*) then (externalName *(optional)* then calcType *(optional)* then hierarchies *(optional)* then dimensions *(optional)* then datasources *(optional)*))

Parent elements

folder , namespace

canGroup

When set to `true`, `canGroup` indicates that query items referenced by the determinant are grouped by the determinant key. This element is always initialized to `false` during import.

Content model

Content type is boolean.

Parent elements

determinant

canonicalName

Specifies the name by which an object is known in the database or plug-in.

Content model

Content type is string.

Parent elements

function , storedProcedure

cardinality

Number of nodes, members, or categories in a hierarchy. Used by OQP to assign solve orders to expressions.

Content model

Content type is nonNegativeInteger.

Parent elements

hierarchy

catalog

A hierarchical element that can represent different information for different databases or that may not be used at all. For example, if the database is SQL Server, the element contains the name of the database; if the database is Oracle, it is not used.

The text may contain macro substitutions. The syntax is defined in the *IBM Cognos Framework Manager User Guide*.

For SAP BW data sources, this is the catalog name returned by MDDataProviderBW::GetCatalogs.

Content model

Mixed content.

Parent elements

dataSource

cmDataSource

Specifies the name of the `dataSource` in Content Manager. Parameters may be set.

The text may contain macro substitutions. The syntax is defined in the *IBM Cognos Framework Manager User Guide*.

Content model

Mixed content.

Parent elements

dataSource

cmSearchPath

The Content Manager search path (Access Manager ID) for an object.

Content model

Content type is string.

Parent elements

basedOn , securityObject

collationSequenceLevel

Cached for UDA only. This element is used only for sortable character data in a dbQuery.

Content model

Content type is integer.

Parent elements

measure , queryItem

collationSequenceName

Cached for UDA only. This element is used only for sortable character data in a dbQuery.

Content model

Content type is string.

Parent elements

measure , queryItem

column

Defines a column in the specified database.

The text may contain macro substitutions. The syntax is defined in the *IBM Cognos Framework Manager User Guide*.

Content model

Mixed content.

Parent elements

sql

comment

Model developer comment. This property is only used by the IBM Cognos Framework Manager application.

Content model

Content type is string.

Parent elements

calculation , dimension , filter , folder , function , hierarchy , hierarchyFolder , level , measure , measureFolder , namespace , object , package , queryItem , queryItemFolder , querySubject , shortcut , updateSubject

conformanceRef

References another query item that represents a column with the same data. Property is only added when planning for SAP extract.

Content model

Content type is string.

Parent elements

calculation , measure , queryItem

connection

Reserved.

Attributes

name

Specifies a name for the property.

Usage: `required` Type: `string`

value

Specifies the value of the parent element.

Usage: `required` Type: `string`

Content model

Empty element.

Parent elements

source

connectionString

Defines the connection string in the form required for UDA, to a data source. If defined, this string takes precedence over any string defined in Content Manager.

Content model

Content type is string.

Parent elements

dataSource

content

Reserved.

Content model

Empty element.

Parent elements

physicalSource

cube

A hierarchical element that contains the definition of dimensions/hierarchies/facts and a collection of associated transaction values at the intersections of the values/members from the dimensions/hierarchies.

Content model

Content type is string.

Parent elements

dataSource

cubeCreatedOn

The date and time that the cube was created.

Content model

Content type is dateTime.

Parent elements

dataSource

cubeCurrentPeriod

The current period for the cube.

Content model

Content type is string.

Parent elements

dataSource

cubeDataUpdatedOn

The date and time that data in the cube was last updated.

Content model

Content type is dateTime.

Parent elements

dataSource

cubeDefaultMeasure

The name of the default measure for the cube.

Content model

Content type is string.

Parent elements

dataSource

cubeDescription

The description of the cube.

Content model

Content type is string.

Parent elements

dataSource

cubeIsOptimized

Returns true if the cube is optimized.

Content model

Content type is boolean.

Parent elements

dataSource

cubePath

The full qualified path to the cube.

Content model

Content type is string.

Parent elements

dataSource

cubeSchemaUpdatedOn

The date and time that the cube schema was last updated.

Content model

Content type is dateTime.

Parent elements

dataSource

currency

The ISO currency code. This element should be present only if the data is currency. If the element is present but empty, the currency code is unknown at modeling time, and must be determined by the data at execution time (currently not supported for relational data sources).

This object is not relevant for SAP BW data sources. SAP BW returns currency information for each value of a fact or measure.

Content model

Content type is string.

Parent elements

calculation , measure , queryItem

dataSource

Specifies a data source for a project.

Content model

(name then guid *(optional)* then property *(any number)*) then (queryProcessing *(optional)* then rollupProcessing *(optional)* then transactionAccessMode *(optional)* then transactionStatementMode *(optional)* then nullValueSorting *(optional)* then cmDataSource then catalog *(optional)* then cube *(optional)* then schema *(optional)* then type then connectionString *(optional)* then aliasTableMapRef *(optional)* then cubeDescription *(optional)* then cubePath *(optional)* then cubeCreatedOn *(optional)* then cubeDataUpdatedOn *(optional)* then cubeSchemaUpdatedOn *(optional)* then cubeIsOptimized *(optional)* then cubeDefaultMeasure *(optional)* then cubeCurrentPeriod *(optional)* then suppression *(optional)* then attributeDimensionsAsProperties *(optional)*)

Parent elements

dataSources

dataSourceRef

References the data source object.

Content model

Content type is string.

Parent elements

datasources , function , sources , storedProcedure

dataSources

A container element that groups `dataSource` elements.

Content model

(dataSource) *(any number)*

Parent elements

project

datasources

References the datasource object for calculations that represent substitution variables (also referred to as macros in the OLAP import doc).

Content model

dataSourceRef *(one or more)*

Parent elements

calculation

datatype

Specifies the datatype for the parent object.

The supported datatypes for SAP BW data sources are: `float`, `character`, `date`, and `time`.

Content model

Content type is NMTOKEN.

The possible values of this element are restricted to the following.

unknown

Datatype is not known.

unsupported

Datatype is not supported.

int16

Datatype is 16 bit integer.

int32

Datatype is 32 bit integer.

int64

Datatype is 64 bit integer.

decimal

Datatype is decimal.

numeric

Datatype is numeric.

float

Datatype is floating point.

float32

Datatype is 32 bit floating point.

float64

Datatype is 64 bit floating point.

binary

Datatype is binary.

binaryLength16

Datatype is 16 bit binary.

date

Datatype is date format (YYMMDD).

time

Datatype is time format (HHMMSS).

dateTime

Datatype is date/time format.

timeInterval

Datatype represents a time interval.

character

Datatype is character.

characterLength16

Datatype is a 16 bit character.

characterLength32

Datatype is a 32 bit character.

blob

Datatype is blob.

textBlob

Datatype is blob, containing text.

blobArray

Datatype is blob, containing an array.

databaseKey

Datatype represents a database key.

varBit

Datatype is variable bit.

bit

Datatype is bit.

boolean

Datatype is boolean.

timeTZ

Datatype represents time, that includes the time zone.

timeStampTZ

Datatype represents a time stamp, that includes the time zone.

intervalYM

Datatype represents a year and month interval.

nChar

Datatype is character, with a length of n.

nVarChar

Datatype is character, with a maximum variable length of n.

Parent elements

calculation , measure , procParameter , procParameter , queryItem , result

dbQuery

Defines a query against the specified database.

Not supported for multidimensional data sources, including SAP BW.

Attributes

multiDb

Specifies that the database definition references multiple databases.

Usage: `optional` Default: `false` Type: `boolean`

Content model

sources then generateSQL *(optional)* then sql then key *(any number)* then index *(any number)* then filters *(optional)* then tableType *(optional)*

Parent elements

definition

decisionRole

Reserved.

Content model

(securityObject) *(any number)*

Parent elements

access , adminAccess

defaultHierarchy

References the default hierarchy for this dimension.

Content model

Content type is string.

Parent elements

dimension

defaultLocale

Specifies the default locale for a project.

Content model

Content type is language.

Parent elements

project

defaultValue

Specifies the value to use if no key match is found. If not set, external sources for `parameterMaps` of this name may be searched.

Content model

Content type is string.

Parent elements

parameterMap

definition

A container for the specified child element. When the parent element is a `dimension` or `querySubject`, the `definition` element can contain a `dbQuery`, `mdQuery`, `modelQuery`, `storedProcedure` or `queryOperation` description.

Content model

dbQuery or mdQuery or modelQuery or storedProcedure or queryOperation

Parent elements

dimension , querySubject , updateSubject

definition

A container for the specified child element. When the parent element is a `securityView`, the `definition` element describes a set.

Content model

set *(any number)*

Parent elements

securityView

definition

A container for the specified child element. When the parent element is a package, the definition element describes a view of the model.

Content model

viewref *(any number)*

Parent elements

package

description

Provides a text-based description for the parent element in the specified language.

When the description applies to the `qosOverride` element, a multilingual string of additional description text is included in the expression editor tips window.

Attributes

locale

Specifies the locale that applies to the parent object.

Usage: `required` Type: `language`

Content model

Content type is string.

Parent elements

calculation , dimension , filter , folder , function , hierarchy , hierarchyFolder , level , measure , measureFolder , namespace , object , package , qosOverride , queryItem , queryItemFolder , querySubject , shortcut , updateSubject

determinant

Captures the relationship between a key and the attributes uniquely described by the key.

Content model

name then key then attributes then canGroup then identifiesRow then property *(any number)*

Parent elements

determinants

determinants

A container element that groups `determinant` elements.

Content model

determinant *(any number)*

Parent elements

querySubject

dimension

A dimension is defined as `regular`, `measure`, `time`, or `currency` using the `type` element.

Attributes

status

Specifies the status of the execution of a query.

Usage: `optional` Default: `valid` Type: `NMTOKEN`.

Attribute values are restricted to the following.

valid

Specifies that the syntax of the object is valid.

invalid

Specifies that the syntax of the object is not valid.

needsReevaluation

Specifies that the object syntax needs to be reviewed.

Content model

(name *(one or more)* then description *(any number)* then (guid *(optional)* then lastChanged *(optional)* then lastChangedBy *(optional)* then comment *(optional)*) then screenTip *(any number)* then property *(any number)*) then ((definition then (previewFilters *(optional)* then securityFilters *(optional)*) then externalizeMethod *(optional)* then externalizeAutoSummary *(optional)* then useInJoinPath *(optional)* then orderOfMagnitude *(optional)* then querySubjectUsage *(optional)* then useLocalCache *(optional)* then (numberOfRows *(optional)*) *(optional)*) *(optional)*) (type then membersRollup *(optional)* then sortMembersMetadata *(optional)* then sortMembersData *(optional)* then sortMembersAndEnableMrf *(optional)* then aliasTableMapRef *(optional)*) then ((measure or measureFolder or (queryItem or queryItemFolder)) *(one or more)* or (defaultHierarchy *(optional)* then (hierarchy or hierarchyFolder) *(one or more)*))

Parent elements

folder , namespace

dimensionRef

References the dimension to which the aggregation rule applies.

Content model

Empty element.

Parent elements

aggregateRule

dimensions

List of dimension references that the set calculation represents. This element is created during an OLAP import.

Content model

refobj *(one or more)*

Parent elements

calculation

displayName

Specifies the name that is displayed to the user for the parent object.

Content model

Content type is string.

Parent elements

filterDefinition , previewFilter , securityFilterDefinition

displayPath

Defines the search path in displayable form.

Content model

Content type is string.

Parent elements

securityObject

displayType

The display type for the query item.

By default, the display type for all SAP BW query items and calculations is set to `value`; its value may also be set to `link`, if applicable.

Content model

Content type is NMTOKEN.

The possible values of this element are restricted to the following.

picture

Specifies that the object is displayed as a graphic.

link

Specifies that the object is displayed as a hypertext link.

value

Specifies that the object is displayed as its return value.

Parent elements

calculation , measure , queryItem

duplicates

Determines whether or not duplicates are removed or preserved in the queryOperation processing.

Content model

Content type is NMTOKEN.

The possible values of this element are restricted to the following.

remove

Specifies that duplicates are removed in a queryOperation.

preserve

Specifies that duplicates are preserved in a queryOperation.

Parent elements

queryOperation

embeddedRelationship

Reserved.

Content model

expression then sql *(optional)* then left then right then joinFilterType *(optional)*

Parent elements

relationshipDefinition

expression

Defines a filter or a calculation.

Content model

Mixed content.

(functionref or refobj or refobjViaShortcut) *(any number)*

Parent elements

calculation , embeddedRelationship , filter , filterDefinition , measure , previewFilter , queryItem , relationship , securityFilterDefinition

externalizeAutoSummary

If `true`, applies the value of the `regularAggregate` element to query items whose usage is `fact`. All fact query items in the parent element are affected.

For example, if `externalizeAutoSummary` is `true` for a `querySubject` and the `regularAggregate` is `count`, the output from the fact query items will only be a count of the queryitems in the query.

Content model

Content type is boolean.

Parent elements

dimension , querySubject , updateSubject

externalizeMethod

Specifies the external form for a query subject. For example, if set to `csv`, a csv file is generated that contains the entire result set of a query subject. Externalization can only be triggered during a publish operation.

Content model

Content type is NMTOKEN.

The possible values of this element are restricted to the following.

default

Specifies that the method used will be the default. The meaning of the default method depends on the type of query subject. For relational data source query subjects, model query subjects whose data source is relational, and stored procedure query subjects, default means embedded. For multidimensional data source query subjects and model query subjects whose data source is multidimensional, default means they will not be externalized.

csv

Use the csv method to generate a comma separated file that contains the results of the query subject. In a csv file, the first row represents the column names and each of the following rows contains one record from the query result set. One file is generated for each query subject that is set to be externalized. With the csv method, you can use locally processed functions to create a dataset for use in Transformer. The generated file contains data based on the native encoding of the current Microsoft

Windows operating system. This option is intended for use only with IBM Cognos Transformer. For any other purpose, use the tab method.

To externalize a SAP BW query subject, use the `csv` option.

tab

Use the tab method to generate a tab delimited file that contains the results of the query subject. The generated file can be used directly as a data source. The generated file contains data based on Unicode using UTF-16 LE (Little Endian) encoding with BOM (Byte Order Mark). One file is generated for each query subject that is set to be externalized.

iqd

Use the iqd method to generate a query definition file for use in Transformer. Native SQL is generated in the model.xml file as a custom property. Cognos SQL is not included. One file is generated for each query subject that is set to be externalized. The generated file contains data based on the native encoding of the current operating system. For example, for Windows 2000, this is specified by the default system locale in the Windows operating system regional settings. The query subject must not require any local processing. It must be able to be run entirely on the data server. You can test the query subject by setting the query processing for this data source to database only. An error message then appears if the query subject requires local processing. If you need to use locally processed functions to create a dataset, use the csv method.

Parent elements

dimension , querySubject , updateSubject

externalName

The `externalName` identifies this object in the external data source. It specifies the name of the underlying object. For example, for relational sources, the `externalName` may be the column name or alias.

For a calculation, a value for `externalName` indicates that the object was created during an import. The expression is set to read-only in the UI.

For OLAP data sources, the value of the column name depends on the underlying object. The following table shows the source of column name values.

Table 6. Column name sources

Element	Source of Column Name
level	Unique name of the level as obtained from MDDataProviderBW::GetLevels
hierarchy	Unique name of the hierarchy as obtained from MDDataProviderBW::GetHierarchies
query item representing a hierarchy level	[MEMBER_UNIQUE_NAME]
level property	Unique property name as obtained from MDDataProviderBW::GetProperties
measure	Unique measure name as obtained from MDDataProviderBW::GetMeasures

Content model

Content type is string.

Parent elements

calculation , hierarchy , level , measure , queryItem

externalNumberOfLevels

Number of levels in a hierarchy. This property is hidden. It is used by OQP to determine which levels are included in the model when constructing the query specification for the RootMembers function.

Content model

Content type is nonNegativeInteger.

Parent elements

hierarchy

externalOrdinal

Represents the identifier for this level or hierarchy. For example, month or number.

For SAP BW, the level or hierarchy number is obtained from MDDataProviderBW::GetLevels.

Content model

Content type is nonNegativeInteger.

Parent elements

level

filePath

Reserved.

Content model

Empty element.

Parent elements

physicalSource

filter

Specifies an expression used to query the data in the specified database.

The result type of a filter expression must be boolean.

Attributes

status

Specifies the status of the resolution of a filter.

Usage: `optional` Default: `valid` Type: `NMTOKEN`.

Attribute values are restricted to the following.

valid

Specifies that the syntax of the object is valid.

invalid

Specifies that the syntax of the object is not valid.

needsReevaluation

Specifies that the object syntax needs to be reviewed.

Content model

(name *(one or more)* then description *(any number)* then (guid *(optional)* then lastChanged *(optional)* then lastChangedBy *(optional)* then comment *(optional)*) then screenTip *(any number)* then property *(any number)*) then (expression)

Parent elements

folder , namespace

filterDefinition

Specifies the definition of a filter to be applied to the parent object.

Attributes

application

Specifies whether the filter is applied before or after auto aggregation.

Usage: `optional` Default: `beforeAutoAggregation` Type: `NMTOKEN`.

Attribute values are restricted to the following.

beforeAutoAggregation

The detail filter that is generated matches the filter expression that you created.

afterAutoAggregation

The detail filter that is generated uses the aggregate of the filter expression that you created.

apply

Specifies the frequency with which the definition of the filter is applied.

Usage: `optional` Default: `always` Type: `NMTOKEN`.

Attribute values are restricted to the following.

always

Specifies that the filter is always applied.

asNeeded

When the filter contains a prompt, allows the user to choose not to supply a value, in which case the filter is not applied.

Content model

refobj or (displayName *(optional)* then expression)

Parent elements

filters

filters

Represents an arbitrary mix of conditions and references to filters.

Content model

(filterDefinition) *(any number)*

Parent elements

dbQuery , mdQuery , modelQuery , queryOperation

fixIdsToDefaultLocale

Object ids that are saved in reports are based on the default locale. When this element is set to `true`, object ids are fixed to use the id in the default locale, regardless of the user locale. When set to `false`, the object ids are always returned in the users' locale. The default is `false`.

Content model

Content type is boolean.

Parent elements

project

folder

A container that does not participate in object references.

Content model

(name *(one or more)* then description *(any number)* then (guid *(optional)* then lastChanged *(optional)* then lastChangedBy *(optional)* then comment *(optional)*) then screenTip *(any number)* then property *(any number)*) then ((folder or function or namespace or object or shortcut or querySubject or updateSubject or relationship or relationshipShortcut or filter or calculation or dimension or scopeRelationship) *(any number)*)

Parent elements

folder , namespace

format

Specifies the format for the parent object.

Content model

Content type is string.

Parent elements

calculation , measure , queryItem

freshness

Specifies the number of seconds to wait before reloading a parameter map during macro evaluation. A value of -1, which is the default value, specifies that the parameter map not be reloaded. A value of 0 specifies that the parameter map be reloaded each time it is being accessed. This property applies to Dynamic Query Mode (DQM) only.

Content model

Content type is string.

Parent elements

parameterMap

function

A function that is stored in a database instance, as specified by its schema. This element is sometimes called a user-defined database function. Contrast this with functions that are built in by the database vendor.

As there is no provision in SAP BW 3.0B for user defined functions, this object should not be used in a model for an SAP BW data source.

Content model

(name *(one or more)* then description *(any number)* then (guid *(optional)* then lastChanged *(optional)* then lastChangedBy *(optional)* then comment *(optional)*) then screenTip *(any number)* then property *(any number)*) then (syntaxTip *(any number)* then canonicalName then dataSourceRef *(optional)* then result then procParameters)

Parent elements

folder , namespace

functionId

An id that identifies a function. This is the same as the id saved in Function Description Service (FDS).

Content model

Content type is string.

Parent elements

qosOverride

functionref

References a function.

Reserved.

Content model

Content type is string.

Parent elements

expression

functionSet

The name of a function set as defined in the Function Description Service (FDS). To view or edit the list of available function sets, refer to the FileList.xml file, located in the *c10_location*\configuration\functions directory on the server where IBM Cognos Framework Manager is installed.

Content model

functionSetID

Parent elements

functionSets

functionSetID

References an FDS function set.

Defines the function set that applies to a data source. Used in the initial population of the functionSets of a securityView when a package is created. To view or edit the list of available function sets, refer to the FileList.xml file, located in the *c10_location*\configuration\functions directory on the server where IBM Cognos Framework Manager is installed.

For SAP BW data sources, set the value to V_SAPBW.

Content model

Content type is NMTOKEN.

Parent elements

functionSet , type

functionSets

Defines the function sets available to report authors. Typically, they are the same as those of the data sources used.

Content model

functionSet *(any number)*

Parent elements

securityView

generateSQL

Causes a query to generate the SQL commands used. If set to `minimized`, only the required query items appear in the generated SQL. If set to `asView`, all query items appear.

Content model

Content type is NMTOKEN.

The possible values of this element are restricted to the following.

asView

Specifies that all query items appear in the generated SQL.

minimized

Specifies that only the required query items appear in the generated SQL.

Parent elements

dbQuery , modelQuery

guid

A unique identifier for the parent object.

Content model

Empty element.

Parent elements

calculation , dataSource , dimension , filter , folder , function , hierarchy , hierarchyFolder , level , measure , measureFolder , namespace , object , package , parameterMap , project , queryItem , queryItemFolder , querySubject , relationship , relationshipShortcut , scopeRelationship , securityView , shortcut , updateSubject

hidden

If set to `true`, the object is hidden in all published packages.

For SAP BW data sources, the folder that contains the surrogate keys should have its hidden property set to true.

Content model

Content type is boolean.

Parent elements

calculation , level , measure , measureFolder , queryItem , queryItemFolder

hierarchies

A list of dimension hierarchies references the set calculation it represents.

Created during an OLAP import.

Content model

refobj *(one or more)*

Parent elements

calculation

hierarchy

A hierarchy within a dimension.

The `rootCaption`, `rootMember` and `rootMUN` properties only apply if the hierarchy contains a single root member. These properties are not set for multi-root hierarchies.

Content model

(name *(one or more)* then description *(any number)* then (guid *(optional)* then lastChanged *(optional)* then lastChangedBy *(optional)* then comment *(optional)*) then screenTip *(any number)* then property *(any number)*) then (object *(any number)*) ((level *(one or more)*) or (queryItem or queryItemFolder) *(one or more)*) then (externalName *(optional)* then multiRoot *(optional)* then balanced *(optional)* then ragged *(optional)* then (rootMember *(optional)* then rootMUN *(optional)* then rootCaption *(any number)*) then sortedHierarchy *(optional)* then cardinality *(optional)* then parentChild *(optional)* then externalNumberOfLevels *(optional)* then isWideFan *(optional)*)

Parent elements

dimension , hierarchyFolder

hierarchyFolder

A folder containing only hierarchies or other hierarchy folders.

Content model

(name *(one or more)* then description *(any number)* then (guid *(optional)* then lastChanged *(optional)* then lastChangedBy *(optional)* then comment *(optional)*) then screenTip *(any number)* then property *(any number)*) then (hierarchy or hierarchyFolder) *(one or more)*

Parent elements

dimension , hierarchyFolder

identifiesRow

When set to `true`, it means the key can uniquely identify a row.

Content model

Content type is boolean.

Parent elements

determinant

index

For compatibility with earlier versions. Do not use.

This functionality has been replaced by determinants.

Content model

unique then queryItems_collection

Parent elements

dbQuery

interface

Reserved. This element is maintained by the application.

Note: For SAP BW data sources, set the value to BW.

Content model

Content type is string.

Parent elements

type

isAccessToNullSuppressionOptionsAllowed

When set to `true`, package users can control the types of values to be suppressed. When set to `false`, users cannot control the types of values to be suppressed. All types of null values (zero values, divide by zero values, missing values, and overflow values) are suppressed when the IBM Cognos studio user invokes suppression. The default is `true`.

If this element is `true`, the `isNullSuppressionAllowed` element must also be`true`.

Content model

Content type is boolean.

Parent elements

package , project

isHierarchical

When set to `true`, indicates that this measure contains other measures. This element is used to optimize runtime performance.

Content model

Content type is boolean.

Parent elements

measure

isManual

When set to `true`, it indicates that the members are unique within the dimension. When used in conjunction with the key element, it means that the query item collection forms the unique key for the level.

Content model

Content type is boolean.

Parent elements

level

isMultiEdgeNullSuppressionAllowed

When set to `true`, package users can select multi-edge or single edge suppression. When set to `false`, users will only have access to single edge suppression. The default is `true`.

If this element is `true`, the `isNullSuppressionAllowed` element must also be `true`.

Content model

Content type is boolean.

Parent elements

package , project

isNullSuppressionAllowed

When set to `true`, suppression is available to package users. When set to `false`, suppression is not available in the published package. The default is `true`.

Content model

Content type is boolean.

Parent elements

package , project

isUnique

When set to `true`, it indicates that the members can be uniquely identified with the businessKey attribute without the additional context from the parent members.

Content model

Content type is boolean.

Parent elements

level

isWideFan

When set to `true`, it indicates that one or more members may have a large number of child members. In some cases, this will trigger an optimization that may generate multiple smaller queries to populate a dynamic cube.

Content model

Content type is boolean.

Parent elements

hierarchy

joinFilterType

Improves the performance of a join by filtering one side of the join with the values that are retrieved by the other side.

In IBM Cognos Analytics versions 11.1.6 and 11.1.7 a filter optimization error might occur. For more information, see "XQE-PLN-0355 filter join optimization error" in the *Framework Manager User Guide*.

Content model

Content type is NMTOKEN.

The possible values of this element are restricted to the following.

none

No optimization is carried out.

in

The generated filter is an `IN` predicate that is composed of constant values of the join keys from the "one-side" join operand.

between

The generated filter is a `BETWEEN` predicate that is composed of the minimum and maximum of the join key values from the "one-side" operand.

table

The generated filter is a table value constructor form of the `IN` predicate.

Parent elements

embeddedRelationship , relationship

key

For compatibility with earlier versions. Do not use.

This functionality has been replaced by determinants.

Content model

name then queryItems_collection

Parent elements

dbQuery

key

Defines the unique identifier for the object.

Content model

Content type is string.

Parent elements

parameterMapEntry

key

The key is a reference to one or more query items that uniquely identifies the referenced attributes within the query subject.

Content model

refobj *(any number)*

Parent elements

determinant

keyRef

When a `parameterMap` element has been defined, the query looks for this unique identifier when it executes and returns the value in the `valueRef` element.

Content model

Empty element.

Parent elements

queryItemMap

lastChanged

Specifies the date and time of the most recent change to the parent object.

Content model

Content type is dateTime.

Parent elements

calculation , dimension , filter , folder , function , hierarchy , hierarchyFolder , level , measure , measureFolder , namespace , object , package , queryItem , queryItemFolder , querySubject , shortcut , updateSubject

lastChangedBy

User name of the last user to change the object.

Content model

Content type is string.

Parent elements

calculation , dimension , filter , folder , function , hierarchy , hierarchyFolder , level , measure , measureFolder , namespace , object , package , queryItem , queryItemFolder , querySubject , shortcut , updateSubject

lastPublished

Defines the last date and time that the package was published to Content Manager.

Content model

Content type is dateTime.

Parent elements

package

lastPublishedCMPath

Last Content Manager path used when the package was published.

Content model

Content type is string.

Parent elements

package

left

References the left object in a relationship.

Content model

refobj

Parent elements

relationshipShortcut

left

References either a measure or regular dimension.

Content model

refobj then mincard then maxcard

Parent elements

scopeRelationship

left

References the left object in a relationship.

Content model

refobj then mincard then maxcard

Parent elements

embeddedRelationship , relationship

level

A level within a hierarchy. Each query item under a level represents a level attribute and/or a key.

Content model

(name *(one or more)* then description *(any number)* then (guid *(optional)* then lastChanged *(optional)* then lastChangedBy *(optional)* then comment *(optional)*) then screenTip *(any number)* then property *(any number)*) then (hidden *(optional)* then (queryItem or queryItemFolder) *(any number)*) isUnique *(optional)* then externalName *(optional)* then externalOrdinal *(optional)* then isManual *(optional)* then memberSort *(optional)* then orderOfMagnitude *(optional)*

Parent elements

hierarchy

levelRef

References a level within a dimension.

Content model

Content type is string.

Parent elements

scope

linkedNode

Reserved.

Attributes

type

Specifies that the property is of the indicated type.

Usage: `required` Type: `string`

Content model

source then queryPath

Parent elements

namespace

loadAsNeeded

This property only applies to parameter maps based on query items. When set to `true`, the parameter map is only partially loaded based on the key that is requested. When set to `false`, the entire parameter map for all the keys is loaded when the first parameter map entry is requested.

Content model

Content type is boolean.

Parent elements

parameterMap

locale

Represents a locale that is available for reports based on a package or a project. Do not add a locale to this list until all objects have been translated into the language for the locale.

Content model

Content type is language.

Parent elements

locales

locales

A container element that groups all locale elements for a package or a project.

This list cannot be independently edited. There must be exactly one instance of every localizable property on every object for each locale in this list.

Content model

locale *(one or more)*

Parent elements

package , project

maxcard

The maximum number (cardinality) of rows that will match this end of the relationship. The value is normally one or many.

Content model

Content type is NMTOKEN.

The possible values of this element are restricted to the following.

unknown

Specifies that the maximum number of rows is not known.

unsupported

Specifies that this feature is not supported.

zero

Specifies that this object has no rows.

one

Specifies that this object has one row only.

many

Specifies that this object can have many rows.

Parent elements

left , left , right , right

maxVersions

The maximum number of published versions of a package to be stored in Content Manager. If this element is absent, versioning is disabled. The value zero (0) means unlimited versions.

Content model

Content type is nonNegativeInteger.

Parent elements

package

mdDimension

A representation of the data components that reflect specific business structures. Typically, a dimension is a nested representation of a business concept.

For OLAP sources, this object is the unique name of the dimension as obtained from MDDataProviderBW::GetDimensions.

Content model

Content type is string.

Parent elements

mdQuery

mdQuery

A query for a multidimensional data source.

An mdQuery object provides additional metadata required for modeling multidimensional query subjects.

Content model

sources then mdDimension *(optional)* then filters *(optional)*

Parent elements

definition

measure

An item that represents a fact in a measure dimension.

Content model

(name *(one or more)* then description *(any number)* then (guid *(optional)* then lastChanged *(optional)* then lastChangedBy *(optional)* then comment *(optional)*) then screenTip *(any number)* then property *(any number)*) then (((expression or externalName) then (hidden *(optional)* then usage *(optional)* then format *(optional)* then currency *(optional)* then (datatype *(optional)* then precision *(optional)* then scale *(optional)* then size *(optional)* then nullable *(optional)* then aggregationRule *(optional)*) then displayType *(optional)* then MIMEType *(optional)* then promptInfo *(optional)* then regularAggregate *(optional)* then semiAggregate *(optional)* then sortOnRef *(optional)* then unSortable *(optional)* then roles *(optional)* then conformanceRef *(optional)*) then (collationSequenceName *(optional)* then collationSequenceLevel *(optional)* then originalCollationSequenceName *(optional)* then originalEncodingName *(optional)*)) *(optional)*) measure *(any number)* then isHierarchical *(optional)* then aggregateRules *(optional)* then allocationRule *(optional)*

Parent elements

dimension , measure , measureFolder

measureFolder

A folder containing only measures or other measure folders.

Content model

(name *(one or more)* then description *(any number)* then (guid *(optional)* then lastChanged *(optional)* then lastChangedBy *(optional)* then comment *(optional)*) then screenTip *(any number)* then property *(any number)*) then (hidden *(optional)* then (measure or measureFolder) *(any number)*)

Parent elements

dimension , measureFolder

measureScope

References a measure to which the measure scope applies.

Attributes

excluded

If set to true, the measure is excluded from the hierarchy implied in the levelRef property of the scope element.

Usage: `optional` Default: `false` Type: `boolean`

Content model

allocation *(optional)* then refobj

Parent elements

scope

memberSort

An ordered list of items that describes how the members will be sorted under a level. The first item is the major sort key. Subsequent items are progressively minor sort keys.

Content model

sortItem *(any number)*

Parent elements

level

membersRollup

When set to `true`, this property indicates that all measure values for members can be rolled up using the aggregation of the measure. If the value is `false`, the measure values may be computed differently for some or all members.

Content model

Content type is boolean.

Parent elements

dimension

MIMEType

Used only when the displayType element is set to `picture` or `link`. The default is context dependent.

Content model

Content type is string.

Parent elements

calculation , measure , queryItem

mincard

The minimum number (cardinality) of rows that will match this end of the relationship. Normally zero or one.

Content model

Content type is NMTOKEN.

The possible values of this element are restricted to the following.

unknown

Specifies that the maximum number of rows is not known.

unsupported

Specifies that this feature is not supported.

zero

Specifies that this object has no rows.

one

Specifies that this object has one row only.

many

Specifies that this object can have many rows.

Parent elements

left , left , right , right

mode

Specifies whether the parameter passed to a procedure is an input parameter, an output parameter, or an input parameter that is changed during execution.

Content model

Content type is NMTOKEN.

The possible values of this element are restricted to the following.

in

Specifies an input parameter.

out

Specifies an output parameter.

inout

Specifies a parameter that is overwritten during execution.

Parent elements

procParameter , procParameter

modelQuery

A query that references metadata in the model. It does not directly reference a data source.

Content model

generateSQL *(optional)* then sql then relationships *(optional)* then filters *(optional)*

Parent elements

definition

mproperty

Represents a multilingual customizable property.

Attributes

locale

Specifies the locale that applies to the parent object.

Usage: `required` Type: `language`

name

Specifies a name for the property.

Usage: `required` Type: `string`

type

Specifies a type for the property.

Usage: `optional` Type: `NMTOKEN`.

Attribute values are restricted to the following.

blob

Specifies the type for this object as blob.

boolean

Specifies the type for this object as boolean.

composite

Specifies that the type for this object is composed of a combination of types.

dateTime

Datatype is date/time format.

enumeration

Specifies that this object is an enumeration.

hidden

Specifies that this object is hidden.

number

Specifies the type for this object as numeric.

readOnly

Specifies that this object is read-only.

ref

Specifies that this object is a reference.

string

Specifies the type for this object as string.

Content model

Mixed content.

multiRoot

Indicates to OQP if a hierarchy has a single root member, or has multiple members at the highest level. This affects the generated MDX.

Content model

Content type is boolean.

Parent elements

hierarchy

name

Specifies a name for the parent object.

Content model

Content type is string.

Parent elements

dataSource , parameterMap , project , relationship , relationshipShortcut , scopeRelationship , securityView

name

Specifies a name for the parent object.

Attributes

locale

Specifies the locale that applies to the parent object.

Usage: `required` Type: `language`

Content model

Content type is string.

Parent elements

calculation , dimension , filter , folder , function , hierarchy , hierarchyFolder , level , measure , measureFolder , namespace , object , package , queryItem , queryItemFolder , querySubject , role , shortcut , updateSubject

name

Specifies a name for the parent object.

Content model

Content type is token.

Parent elements

key

name

An arbitrary name to identify the determinant. The unilingual name is initially generated during import but you can modify it. Determinants must have a unique name.

This name is only used by the IBM Cognos Framework Manager user interface(FMUI).

Content model

Content type is string.

Parent elements

determinant

namespace

A container that participates in object references. The first part of a reference to any object is its nearest containing namespace (which may be itself).

Each SAP BW data source (for example, InfoCube, Query) must be represented in an IBM Cognos Framework Manager model in a separate namespace.

Content model

(name *(one or more)* then description *(any number)* then (guid *(optional)* then lastChanged *(optional)* then lastChangedBy *(optional)* then comment *(optional)*) then screenTip *(any number)* then property *(any number)*) then ((folder or function or namespace or object or shortcut or querySubject or updateSubject or relationship or relationshipShortcut or filter or calculation or dimension or scopeRelationship) *(any number)*) (linkedNode) *(any number)*

Parent elements

folder , namespace , project

nullable

If `true`, this object can contain NULL values.

For all SAP BW query items this is `false`.

Content model

Content type is boolean.

Parent elements

calculation , measure , procParameter , procParameter , queryItem , result

nullValueSorting

Specifies how to sort null values.

Content model

Content type is NMTOKEN.

The possible values of this element are restricted to the following.

unspecified

Specifies that nulls are sorted using the default database vendor sorting behavior.

ascendingNullsFirst

Specifies that nulls sort first when ascending and sort last when descending.

ascendingNullsLast

Specifies that nulls sort last when ascending and sort first when descending.

nullsFirst

Specifies that nulls appear first in the sort order.

nullsLast

Specifies that nulls appear last in the sort order.

Parent elements

dataSource

numberOfRows

Specifies the number of rows of data for the parent query subject.

Content model

Content type is integer.

Parent elements

dimension , querySubject , updateSubject

object

Represents a generic customizable object.

Content model

(name *(one or more)* then description *(any number)* then (guid *(optional)* then lastChanged *(optional)* then lastChangedBy *(optional)* then comment *(optional)*) then screenTip *(any number)* then property *(any number)*) then (object *(any number)*)

Parent elements

folder , hierarchy , namespace , object

orderOfMagnitude

Specifies the common (base 10) logarithm of the number of rows in a level.

Content model

Content type is integer.

Parent elements

dimension , level , querySubject , updateSubject

originalCollationSequenceName

Cached for UDA only. This element is used only for sortable character data in a `dbQuery`.

Content model

Content type is string.

Parent elements

measure , queryItem

originalEncodingName

Specifies how the data is encoded in the database. Cached for UDA only. This element is used only for sortable character data in a `dbQuery`.

Content model

Content type is string.

Parent elements

measure , queryItem

package

Represents a packaging of model objects to be published for one or more reporting applications.

Attributes

isRoleBased

If true, this package is reserved for specifying object security by role. There is one package for each role for which object security has been set. The package must have exactly one security view that specifies the role. For that role, each object referenced by that security view has visibility opposite to the normal state, in all normal packages published for this model.

Usage: `optional` Default: `false` Type: `boolean`

Content model

(name *(one or more)* then description *(any number)* then (guid *(optional)* then lastChanged *(optional)* then lastChangedBy *(optional)* then comment *(optional)*) then screenTip *(any number)* then property *(any number)*) then (lastPublished *(optional)* then lastPublishedCMPath *(optional)* then maxVersions *(optional)* then locales then definition then adminAccess then isNullSuppressionAllowed *(optional)* then isMultiEdgeNullSuppressionAllowed *(optional)* then isAccessToNullSuppressionOptionsAllowed *(optional)* then useV5DataServer *(optional)*)

Parent elements

packages

packages

A container element that groups `package` elements.

Content model

package *(any number)*

Parent elements

project

parameterMap

A lookup table for controlling query generation. As a minimum, the _env and _governor maps are required.

Attributes

hidden

Specifies whether or not the parameterMap element is hidden.

Usage: `optional` Default: `false` Type: `boolean`

Content model

(name then guid *(optional)* then property *(any number)*) then (defaultValue *(optional)* then freshness *(optional)* then loadAsNeeded *(optional)* then (parameterMapEntry *(one or more)* or queryItemMap) *(optional)*)

Parent elements

parameterMaps

parameterMapEntry

Specifies that the object is an entry in a parameter map.

Content model

key then value

Parent elements

parameterMap

parameterMaps

A container element that groups `parameterMap` elements.

Content model

parameterMap *(one or more)*

Parent elements

project

parameterName

Specifies a name for a parameter passed to, or resulting from, execution of a procedure.

Content model

Content type is string.

Parent elements

procParameter , procParameter

parentChild

Indicates that the hierarchy is a collection of members arranged in a parent-child hierarchy.

Content model

Content type is boolean.

Parent elements

hierarchy

physicalSource

Reserved.

Content model

filePath or content

Parent elements

physicalSources

physicalSources

Reserved.

Content model

(physicalSource) *(any number)*

Parent elements

project

precision

Represents the allowed number of digits or characters. Default is context-dependent. This element is used only for numeric and decimal types.

The precision of numeric values from SAP BW data sources should be set to zero.

Content model

Content type is integer.

Parent elements

calculation , measure , procParameter , procParameter , queryItem , result

previewFilter

A filter that is applied at execution time when the report or model is being used in design mode. It is not applied when the report is run.

Attributes

application

Specifies whether the filter is applied before or after auto aggregation.

Usage: `optional` Default: `beforeAutoAggregation` Type: `NMTOKEN`.

Attribute values are restricted to the following.

beforeAutoAggregation

The detail filter that is generated matches the filter expression that you created.

afterAutoAggregation

The detail filter that is generated uses the aggregate of the filter expression that you created.

Content model

refobj or (displayName *(optional)* then expression)

Parent elements

previewFilters

previewFilters

A container object for `previewFilter` elements.

Content model

(previewFilter) *(any number)*

Parent elements

calculation , dimension , querySubject , updateSubject

procParameter

Defines a parameter required to call a function or stored procedure. This element is used to determine whether an expression that uses the function or procedure is valid. The value of the parameter is supplied from the expression that uses this function or stored procedure.

SAP BW does not support stored procedures and therefore this object should not form part of an IBM Cognos Framework Manager model based on an SAP BW data source.

Content model

parameterName then mode then (datatype *(optional)* then precision *(optional)* then scale *(optional)* then size *(optional)* then nullable *(optional)* then aggregationRule *(optional)*)

Parent elements

procParameters

procParameter

Defines a parameter required to call a stored procedure. This element is used to determine whether an expression that uses the stored procedure is valid. The value of the parameter is supplied from the expression that uses this stored procedure.

OLAP does not support stored procedures and therefore this object should not form part of an IBM Cognos Framework Manager model based on an OLAP data source.

Content model

(parameterName then mode then (datatype *(optional)* then precision *(optional)* then scale *(optional)* then size *(optional)* then nullable *(optional)* then aggregationRule *(optional)*)) then value *(optional)*

Parent elements

procParameters

procParameters

A container object for `procParameter` elements.

Content model

procParameter *(any number)*

Parent elements

function

procParameters

A container object for `procParameter` elements.

Content model

procParameter *(any number)*

Parent elements

storedProcedure

project

The root object of the model. For object referencing purposes, the name of this namespace is empty.

Attributes

containsDynamicContent

Reserved.

Usage: optional Default: false Type: boolean

queryMode

Specifies the query mode of the model.

Usage: optional Default: mixed Type: NMTOKEN.

Attribute values are restricted to the following.

mixed

Specifies that dynamic query mode is used as an option for testing and publishing.

dynamic

Specifies that dynamic query mode is used.

useMFW

Reserved. Enables the model to be opened using the metadata framework.

Usage: optional Default: false Type: NMTOKEN.

Attribute values are restricted to the following.

true

Specifies that the model can be opened using the metadata framework.

false

Specifies that the model be opened using the standard QECL RTM process.

always

Specifies that the model is always opened using the metadata framework.

Content model

(name then guid *(optional)* then property *(any number)*) then ((locales then defaultLocale then fixIdsToDefaultLocale *(optional)* then qosOverrides *(optional)* then isNullSuppressionAllowed *(optional)* then isMultiEdgeNullSuppressionAllowed *(optional)* then isAccessToNullSuppressionOptionsAllowed *(optional)* then steward *(optional)* then physicalSources *(optional)*) then namespace then dataSources then parameterMaps then securityViews then packages)

promptCascadeOnRef

If prompted for, a preceding prompt is generated for each of the referenced query items.

Content model

Content type is string.

Parent elements

promptInfo

promptDisplayItemRef

If prompted for, one or more of the referenced objects becomes the display value of the prompt control.

Content model

Content type is string.

Parent elements

promptInfo

promptFilterItemRef

If automatic filters are generated for this object, they will be created on the referenced objects.

Content model

Content type is string.

Parent elements

promptInfo

promptInfo

A container element for information about a prompt.

Content model

promptType then promptCascadeOnRef *(optional)* then promptDisplayItemRef *(optional)* then promptFilterItemRef *(optional)* then promptUseItemRef *(optional)*

Parent elements

calculation , measure , queryItem

promptType

The type of prompt control to be used when prompting for a value for this item.

For SAP BW, `selectWithSearch`, `selectDateTime`, and `selectInterval` prompt types are not supported. The `selectTime` prompt type should only be used for SAP BW query items (facts) of type `time`.

Content model

Content type is NMTOKEN.

The possible values of this element are restricted to the following.

serverDetermined

Specifies that the prompt type is determined by the server during execution.

editBox

Specifies that the prompt is displayed as an edit box.

selectDate

Specifies that the prompt displays a selectable date.

selectDateTime

Specifies that the prompt displays a selectable date and time.

selectInterval

Specifies that the prompt displays a selectable interval.

selectTime

Specifies that the prompt displays a selectable time.

selectValue

Specifies that the prompt displays a list of values from which the user can select.

selectWithSearch

Specifies that the prompt is displayed as a user-defined search.

Note: This value is not applicable to a `queryItem` where `usage` is set to `fact`.

selectWithTree

Specifies that the prompt is displayed as a tree.

Parent elements

promptInfo

promptUseItemRef

If prompted for, one or more of the referenced objects is used as the prompt value(s).

Content model

Content type is string.

Parent elements

promptInfo

property

Represents a generic customizable property.

Attributes

name

Specifies a name for the property.

Usage: `required` Type: `string`

type

Specifies a type for the property.

Usage: `optional` Type: `NMTOKEN`.

Attribute values are restricted to the following.

blob

Specifies the type for this object as blob.

boolean

Specifies the type for this object as boolean.

composite

Specifies that the type for this object is composed of a combination of types.

dateTime

Datatype is date/time format.

enumeration

Specifies that this object is an enumeration.

hidden

Specifies that this object is hidden.

number

Specifies the type for this object as numeric.

readOnly

Specifies that this object is read-only.

ref

Specifies that this object is a reference.

string

Specifies the type for this object as string.

Content model

Mixed content.

(property) *(any number)*

Parent elements

calculation , dataSource , determinant , dimension , filter , folder , function , hierarchy , hierarchyFolder , level , measure , measureFolder , namespace , object , package , parameterMap , project , property , queryItem , queryItemFolder , querySubject , relationship , relationshipShortcut , scopeRelationship , securityView , shortcut , updateSubject

qosLevel

Quality of service indicator as defined in Function Description Service(FDS).

Content model

Content type is NMTOKEN.

The possible values of this element are restricted to the following.

unavailable

The function is not available to the selected data sources in the project.

restricted

The function is available in the project but may have poor performance with some data sources.

limited

The function is only available to some data sources in the project.

ok

The function has no limitations in the data sources in the project.

Parent elements

qosOverride

qosOverride

A quality of service override indicates when the quality of service level is changed within IBM Cognos Framework Manager.

Content model

functionId then qosLevel *(optional)* then description *(any number)*

Parent elements

qosOverrides

qosOverrides

A container object for `qosOverride` elements.

Content model

qosOverride *(any number)*

Parent elements

project , securityView

queryItem

A reportable data item from a query subject or as an attribute of a dimension. For example, a item in the projection list of the SQL definition of a query subject.

Content model

(name *(one or more)* then description *(any number)* then (guid *(optional)* then lastChanged *(optional)* then lastChangedBy *(optional)* then comment *(optional)*) then screenTip *(any number)* then property *(any number)*) then (((expression or externalName) then (hidden *(optional)* then usage *(optional)* then format *(optional)* then currency *(optional)* then (datatype *(optional)* then precision *(optional)* then scale *(optional)* then size *(optional)* then nullable *(optional)* then aggregationRule *(optional)*) then displayType *(optional)* then MIMEType *(optional)* then promptInfo *(optional)* then regularAggregate *(optional)* then semiAggregate *(optional)* then sortOnRef *(optional)* then unSortable *(optional)* then roles *(optional)* then conformanceRef *(optional)*) then (collationSequenceName *(optional)* then collationSequenceLevel *(optional)* then originalCollationSequenceName *(optional)* then originalEncodingName *(optional)*)) *(optional)*)

Parent elements

dimension , hierarchy , level , queryItemFolder , querySubject

queryItemFolder

A folder containing only query items.

Content model

(name *(one or more)* then description *(any number)* then (guid *(optional)* then lastChanged *(optional)* then lastChangedBy *(optional)* then comment *(optional)*) then screenTip *(any number)* then property *(any number)*) then (hidden *(optional)* then (queryItem or queryItemFolder) *(any number)*)

Parent elements

dimension , hierarchy , level , queryItemFolder , querySubject

queryItemMap

Specifies that the set of entries in the `parameterMap` is based on two `queryItem` references: one represents the set of keys, and the second represents the set of values.

Content model

keyRef then valueRef

Parent elements

parameterMap

queryItems_collection

A list of query items that together uniquely identify row value sets in a query subject.

Content model

refobj *(any number)*

Parent elements

index , key

queryOperation

A query that represents the union or intersection of two query subjects. this does not apply to dimensions.

Content model

querySubjectRefs then setOperation then duplicates then filters *(optional)*

Parent elements

definition

queryPath

Reserved.

Attributes

value

Specifies the value of the parent element.

Usage: `required` Type: `string`

Content model

Empty element.

Parent elements

linkedNode

queryProcessing

Determines what degree of local processing is allowed.

For OLAP data sources in this release, this object is ignored but should be set to `databaseOnly`.

Content model

Content type is NMTOKEN.

The possible values of this element are restricted to the following.

databaseOnly

The database server does all the SQL processing and execution. A run-time error will appear if any part of the query relating to this data source requires local SQL processing (that is, on the report server).

limitedLocal

The database server does as much of the SQL processing and execution as possible. The remainder is done on the report server.

Parent elements

dataSource

querySubject

Represents a view or query of a physical database.

Attributes

status

Specifies the status of the execution of a query.

Usage: `optional` Default: `valid` Type: `NMTOKEN`.

Attribute values are restricted to the following.

valid

Specifies that the syntax of the object is valid.

invalid

Specifies that the syntax of the object is not valid.

needsReevaluation

Specifies that the object syntax needs to be reviewed.

Content model

(name *(one or more)* then description *(any number)* then (guid *(optional)* then lastChanged *(optional)* then lastChangedBy *(optional)* then comment *(optional)*) then screenTip *(any number)* then property *(any number)*) then ((definition then (previewFilters *(optional)* then securityFilters *(optional)*) then externalizeMethod *(optional)* then externalizeAutoSummary *(optional)* then useInJoinPath *(optional)* then orderOfMagnitude *(optional)* then querySubjectUsage *(optional)* then useLocalCache *(optional)* then (numberOfRows *(optional)*) *(optional)*) *(optional)*) determinants *(optional)* then (queryItem or queryItemFolder) *(any number)*

Parent elements

folder , namespace

querySubjectRefs

References the query subjects included in a `queryOperation`.

Content model

refobj

Parent elements

queryOperation

querySubjectUsage

Specifies the usage of the query subject.

Content model

Content type is NMTOKEN.

The possible values of this element are restricted to the following.

automatic

Specifies that the query engine will detect if the query subject is a dimension or a fact.

bridge

Specifies that the query subject refers to a bridge table.

summaryQuery

Specifies that the query subject will be treated as a summary query.

Parent elements

dimension , querySubject , updateSubject

queryType

The type of query model that this data source understands. SQL sources are relational. MDS sources are multidimensional.

For SAP BW data sources, the value is `multidimensional`.

Content model

Content type is NMTOKEN.

The possible values of this element are restricted to the following.

relational

The data source understands a relational query model. For SQL sources, the value is relational.

multidimensional

The data source understands a multidimensional query model. For MDS sources, the value is multidimensional.

Parent elements

type

ragged

Defines the hierarchy as ragged. The only conceptual difference between a ragged and an unbalanced hierarchy is the fact that branches of a ragged hierarchy descend through all existing levels, even levels that do not contain data.

Content model

Content type is boolean.

Parent elements

hierarchy

refobj

References an object.

Content model

Content type is string.

Parent elements

attributes , dimensions , expression , filterDefinition , hierarchies , key , left , left , left , previewFilter , queryItems_collection , refobjViaShortcut , relationshipDefinition , right , right , right , securityFilterDefinition , set , shortcut , sortItem

refobj

References two query subjects to be included in the set.

Content model

Content type is string.

Parent elements

querySubjectRefs

refobj

References a measure or the entire measures dimension. If the reference is to a measure dimension, then it implies that the scope applies to all measures within the dimension.

Content model

Content type is string.

Parent elements

measureScope

refobjViaShortcut

References a query item that goes through a shortcut. The first reference is to a shortcut for a query subject. The second reference is to a query item within that query subject. This form of reference preserves the container context used by the system to choose between join paths that would otherwise be ambiguous.

Attributes

dataItemName

Deprecated.

Usage: `optional` Type: `string`

Content model

refobj then refobj

Parent elements

expression

regularAggregate

Specifies the additive aggregation rule. Unless otherwise specified, zero and NULL values are included in calculating the result.

For SAP BW data sources, may take the values: `unsupported`, `average`, `averageNotZero`, `count`, `countNotZero`, `automatic`, `calculated` (for expressions), `sum`, `maximum`, `minimum`, `median`, `standardDeviation`, `variance`. For a measure query item, its value may be different from what

is defined in SAP BW (with the exception that a redefined aggregate cannot be `averageNotZero` or `countNotZero`).

Content model

Content type is NMTOKEN.

The possible values of this element are restricted to the following.

unsupported

Aggregation is not supported for this object.

unknown

Specifies that the aggregation type is unknown.

average

Aggregation is computed as an average of the values in the result set for this object.

automatic

The aggregation rule is determined automatically at run time. Wherever possible, the value of calculated is used, then sum where necessary, followed by unsupported.

calculated

Aggregated values are computed by aggregating the elements of the expression rather than the result. For example, if the aggregate rule for QI_a is set to sum, and the rule for QI_b is set to average, then if this query item is based on the expression (QI_a / QI_b) * 100, the aggregate is computed as (sum(QI_a) / average(QI_b)) * 100.

count

Aggregation is computed as a count of the items in the result set for this object.

countDistinct

Aggregation is computed as a count of unique items in the result set for this object.

countNonZero

Excludes zero and NULL values.

maximum

Aggregation determines the maximum value of the result set for this object.

median

Aggregation determines the median value of the result set for this object.

minimum

Aggregation determines the minimum value of the result set for this object.

standardDeviation

Aggregation determines the standard deviation for the result set for this object.

sum

Aggregation determines the sum of the values in the result set for this object.

variance

Aggregation determines the variance of the result set for this object.

Parent elements

calculation , measure , queryItem

relationship

An expression that can be used to join two query subjects.

There is no SQL associated with a relationship in a model based on an SAP BW data source.

Attributes

status

Specifies the status of the resolution of a relationship.

Usage: `optional` Default: `valid` Type: `NMTOKEN`.

Attribute values are restricted to the following.

valid

Specifies that the syntax of the object is valid.

invalid

Specifies that the syntax of the object is not valid.

needsReevaluation

Specifies that the object syntax needs to be reviewed.

Content model

(name then guid *(optional)* then property *(any number)*) then (expression then sql *(optional)* then left then right then joinFilterType *(optional)*)

Parent elements

folder , namespace

relationshipDefinition

Defines a relationship.

Content model

refobj or embeddedRelationship

Parent elements

relationships

relationshipRef

References a relationship.

Content model

Content type is string.

Parent elements

relationshipShortcut

relationships

Reserved.

Content model

(relationshipDefinition) *(any number)*

Parent elements

modelQuery

relationshipShortcut

Defines a shortcut reference to a relationship. Used to represent that relationship in a container where one or both ends are represented by shortcuts.

Content model

(name then guid *(optional)* then property *(any number)*) then (left then right then relationshipRef)

Parent elements

folder , namespace

result

Contains the value resulting from the execution of a function.

Content model

datatype *(optional)* then precision *(optional)* then scale *(optional)* then size *(optional)* then nullable *(optional)* then aggregationRule *(optional)*

Parent elements

function

right

References the right object in a relationship.

Content model

refobj

Parent elements

relationshipShortcut

right

References either a measure or regular dimension.

Content model

refobj then mincard then maxcard

Parent elements

scopeRelationship

right

References the right object in a relationship.

Content model

refobj then mincard then maxcard

Parent elements

embeddedRelationship , relationship

role

A multilingual property that indicates the role of a member attribute such as the caption or description. All role names that start with an underscore are reserved names.

Attributes

intrinsic

If set to true, indicates that the attribute with this role will not be displayed in the studios but is available via the roleValue function. If more than one role is specified, the intrinsic attribute is ANDed for all roles.

Usage: `optional` Default: `false` Type: `boolean`

Content model

name *(one or more)*

Parent elements

roles

roles

This property is only applicable for query items that act as attributes for a level or hierarchy.

Content model

role *(any number)*

Parent elements

calculation , measure , queryItem

rollupProcessing

Determines whether aggregate rollups above the lowest level are computed locally or in the database. The default is set to `local` if local query processing is enabled, and is set to `database` otherwise.

Not applicable for SAP BW data sources.

Content model

Content type is NMTOKEN.

The possible values of this element are restricted to the following.

unspecified

The aggregation rollup is not specified.

local

All aggregation rollups are computed locally (in the report server) using a running aggregate (for example, RSUM). Running aggregates spread the cost of this computation as the data is retrieved. Use this option if the local computer has more idle resources than the database computer, or if you find through experiment that it is the fastest method.

database

Aggregation rollups are computed by the underlying database software if possible. Otherwise, they are computed locally (provided local query processing is enabled). Running aggregates are used, but the cost is incurred by the database server instead of the report server. Use this option if the database computer has more idle resources than the local computer, or if you find through experiment that it is the fastest method.

extended

All aggregation rollups are computed by the database server using an extended aggregate (for example, XSUM). Extended aggregates incur the entire cost of this computation up front. Typically, this is the fastest method, but only where the database is set up to take advantage of materialized views. For databases where OLAP functionality is supported, this is translated into the appropriate OLAP aggregate functions.

Parent elements

dataSource

rootCaption

Caption property of the root member. This property is only set if the hierarchy has only a single root member. This is a multilingual property.

Attributes

locale

Specifies the locale that applies to the parent object.

Usage: `required` Type: `language`

Content model

Content type is string.

Parent elements

hierarchy

rootMember

Specifies the external name of the root member for a hierarchy as captured from the data source. This element is only applicable to OLAP sources.

Content model

Content type is string.

Parent elements

hierarchy

rootMUN

Member unique name for the root member. This property is only set if the hierarchy has only a single root member. This is a multilingual property.

Content model

Content type is string.

Parent elements

hierarchy

scale

Defines the number of digits to the right of the implied decimal point. Used only for numeric and decimal types.

Content model

Content type is integer.

Parent elements

calculation , measure , procParameter , procParameter , queryItem , result

schema

A hierarchical element that can represent different information for different databases, or that may not be used at all. For example, for SQL server or Oracle, the element contains the name of the owner.

The text may contain macro substitutions. The syntax is defined in the *IBM Cognos Framework Manager User Guide*.

For SAP BW data sources, assign this object a value of SAP_BW.

Content model

Mixed content.

Parent elements

dataSource

scope

Defines the level to measure scope. To define a `scopeRelationship`, there must be at least one level. If no `scope` element is defined, all measures scope to the leaf level of the dimension.

Content model

levelRef then measureScope *(one or more)*

Parent elements

scopeRelationship

scope

Reserved.

Attributes

name

Specifies a name for the property.

Usage: `required` Type: `string`

value

Specifies the value of the parent element.

Usage: `required` Type: `string`

Content model

Empty element.

Parent elements

source

scopeRelationship

Defines that a dimension is within scope of a measure dimension.

Content model

(name then guid *(optional)* then property *(any number)*) then (left then right then scope *(any number)*)

Parent elements

folder , namespace

screenTip

Defines the text that displays when a user passes the cursor over an interface element. Text should be specific to the language of the locale attribute.

Attributes

locale

Specifies the locale that applies to the parent object.

Usage: `required` Type: `language`

Content model

Content type is string.

Parent elements

calculation , dimension , filter , folder , function , hierarchy , hierarchyFolder , level , measure , measureFolder , namespace , object , package , queryItem , queryItemFolder , querySubject , shortcut , updateSubject

securityFilterDefinition

Defines a filter that determines the security for the parent object in the current session.

Content model

securityObject then basedOn *(optional)* then (refobj or (displayName *(optional)* then expression)) *(optional)*

Parent elements

securityFilters

securityFilters

At execution time, all filters that match the security objects for this session are applied, combined by OR.

Content model

(securityFilterDefinition) *(any number)*

Parent elements

calculation , dimension , querySubject , updateSubject

securityObject

Defines an account, group, or role.

Attributes

type

Specifies that the property is of the indicated type.

Usage: `required` Type: `NMTOKEN`.

Attribute values are restricted to the following.

account

Specifies that the securityObject is an account.

group

Specifies that the securityObject is a group.

role

Specifies that the securityObject is a role.

Content model

displayPath then cmSearchPath

Parent elements

decisionRole , securityFilterDefinition

securityView

A relationship between a subset of the model and whoever has access to it in the published model for reporting purposes.

Not supported for SAP BW data sources in this release.

Attributes

isRoleBased

If true, this package is reserved for specifying object security by role. There is one package for each role for which object security has been set. The package must have exactly one security view that specifies the role. For that role, each object referenced by that security view has visibility opposite to the normal state, in all normal packages published for this model.

Usage: `optional` Default: `false` Type: `boolean`

Content model

(name then guid *(optional)* then property *(any number)*) then ((definition then access then functionSets *(optional)* then qosOverrides *(optional)*) *(optional)*)

Parent elements

securityViews

securityViews

A container for `securityView` elements.

Not applicable for SAP BW data sources in this release.

Content model

securityView *(any number)*

Parent elements

project

semiAggregate

Specifies the semi-additive aggregation rule. Unless otherwise specified, enumeration values are the same as the `regularAggregegage`. This element applies only to OLAP sources and is read-only. It is not supported for relational sources.

For SAP BW, this property should only be assigned to those measures with an exceptional aggregator as specified by MDDataProvider::GetMeasures. Overriding the semi-aggregate rule defined in SAP BW has no effect in the IBM Cognos software.

Content model

Content type is NMTOKEN.

The possible values of this element are restricted to the following.

unsupported

Aggregation is not supported for this object.

unknown

Specifies that the aggregation type is unknown.

average

Aggregation is computed as an average of the values in the result set for this object.

automatic

The aggregation rule is determined automatically at run time. Wherever possible, the value of calculated is used, then sum where necessary, followed by unsupported.

calculated

Aggregated values are computed by aggregating the elements of the expression rather than the result. For example, if the aggregate rule for QI_a is set to sum, and the rule for QI_b is set to average, then if this query item is based on the expression (QI_a / QI_b) * 100, the aggregate is computed as (sum(QI_a) / average(QI_b)) * 100.

count

Aggregation is computed as a count of the items in the result set for this object.

countDistinct

Aggregation is computed as a count of unique items in the result set for this object.

countNonZero

Excludes zero and NULL values.

maximum

Aggregation determines the maximum value of the result set for this object.

median

Aggregation determines the median value of the result set for this object.

minimum

Aggregation determines the minimum value of the result set for this object.

standardDeviation

Aggregation determines the standard deviation for the result set for this object.

sum

Aggregation determines the sum of the values in the result set for this object.

variance

Aggregation determines the variance of the result set for this object.

Parent elements

calculation , measure , queryItem

set

A set of included or excluded objects, or included `securityViews`.

Attributes

includeRule

Specifies whether or not objects in the set are included, excluded or hidden.

Usage: `optional` Default: `include` Type: `NMTOKEN`.

Attribute values are restricted to the following.

include

Specifies that the set represents objects that are to be included.

hide

Specifies that the set represents objects that are to be hidden.

exclude

Specifies that the set represents objects that are to be excluded.

Content model

refobj *(one or more)* or viewref *(one or more)*

Parent elements

definition

setOperation

Identifies the operation that is applied to the query subjects in a `queryOperation`.

Content model

Content type is NMTOKEN.

The possible values of this element are restricted to the following.

UNION

The operation is UNION.

INTERSECT

The operation is INTERSECT.

EXCEPT

The operation is EXCEPT.

Parent elements

queryOperation

shortcut

An object that points to another object. Used to represent an object in another container, or at the other end of a self-join.

Content model

(name *(one or more)* then description *(any number)* then (guid *(optional)* then lastChanged *(optional)* then lastChangedBy *(optional)* then comment *(optional)*) then screenTip *(any number)* then property *(any number)*) then (refobj then targetType then treatAs *(optional)* then useInJoinPath *(optional)*)

Parent elements

folder , namespace

signon

Reserved.

Attributes

name

Specifies a name for the property.

Usage: `required` Type: `string`

value

Specifies the value of the parent element.

Usage: `required` Type: `string`

Content model

Empty element.

Parent elements

source

size

Specifies the maximum number of bytes needed to hold a value.

Content model

Content type is long.

Parent elements

calculation , measure , procParameter , procParameter , queryItem , result

sortedHierarchy

Indicates if the members at each level of a hierarchy are sorted. Used by OQP to optimize query generation in the presence of user-specified sorts.

Content model

Content type is NMTOKEN.

The possible values of this element are restricted to the following.

ascending

Specifies an ascending sort.

descending

Specifies a descending sort.

none

Specifies no sort.

Parent elements

hierarchy

sortItem

In a list of items, the first item is the major sort key. Subsequent items are progressively minor sort keys.

Attributes

nullPlacement

Placement of NULLs, either first, last, or depending on the underlying database. For most databases NULLS LAST is the default for ascending order, and NULLS FIRST is the default for descending order.

Usage: `optional` Default: `nullsLast` Type: `NMTOKEN`.

Attribute values are restricted to the following.

nullsUnspecified

The placement of nulls is not specified.

nullsFirst

Specifies that nulls appear first in the sort order.

nullsLast

Specifies that nulls appear last in the sort order.

sort

Specifies the sort rule. NULL values always appear at the end of a sort.

Usage: `optional` Default: `ascending` Type: `NMTOKEN`.

Attribute values are restricted to the following.

ascending

Specifies an ascending sort.

descending

Specifies a descending sort.

Content model

refobj

Parent elements

memberSort

sortMembersAndEnableMrf

When `true`, indicates that the data is sorted when coming from the source and multiple queries access the underlying relational database. The sort provides member relative functions(MRFs) with a consistent member order.

When `false`, functions that rely on the member order (for example, prevMember, nextMember) will cause an exception. The element should be `false` when using position-sensitive functions.

If this element is `true`, the `sortMembersMetadata` element must also be `true`.

Note: This element is not valid on SAP BW or measure dimensions.

Content model

Content type is boolean.

Parent elements

dimension

sortMembersData

When `true`, indicates that the data will be sorted on the report.

The sort is applied to data items based on level references or expressions requesting members of a level.

This element is not valid on SAP BW or measure dimensions.

Content model

Content type is boolean.

Parent elements

dimension

sortMembersMetadata

When `true`, members will be sorted in the metadata tree.

This element is not valid on SAP BW or measure dimensions.

Content model

Content type is boolean.

Parent elements

dimension

sortOnRef

Specifies that when sorting this item, use the query item referred to as the sort key.

Content model

Content type is string.

Parent elements

calculation , measure , queryItem

source

Reserved.

Attributes

type

Specifies that the property is of the indicated type.

Usage: `required` Type: `string`

Content model

(connection or scope or signon) *(one or more)*

Parent elements

linkedNode

sources

Only one data source is allowed if the SQL type is `native`.

Not applicable to models based on SAP BW data sources.

Content model

(dataSourceRef) *(any number)*

Parent elements

dbQuery , mdQuery

sql

A read-only cache of the SQL that results from planning the model query. It is updated every time the modelQuery changes, and can be used as a backup if the dependant model objects are lost.

Not applicable to models based on SAP BW data sources.

Attributes

type

Specifies the level of flexibility the system is allowed in optimizing the SQL from this element, before presenting it to the data source.

Usage: `optional` Default: `cognos` Type: `NMTOKEN`.

Attribute values are restricted to the following.

cognos

Cognos SQL can be optimized by the system and combined with other SQL before being sent to the data source.

oracle

OBSOLETE

db2

OBSOLETE

sqlServer

OBSOLETE

sql92

OBSOLETE

sql99

OBSOLETE: synonym for native.

native

Native SQL cannot be modified except by macro substitution, but can be combined with other SQL into a single SQL query to send to the data source.

passThrough

Pass-through SQL cannot be modified except by macro substitution, and cannot be combined with other SQL. It must be sent to the data source as a stand-alone query. However, results can be combined with results of other queries.

Content model

Mixed content.

(column or table) *(any number)*

Parent elements

dbQuery , embeddedRelationship , modelQuery , relationship

steward

Specifies the name of the person or process responsible for the project.

Content model

Content type is string.

Parent elements

project

storedProcedure

Specifies the name, data source, parameters, and type of a procedure stored in the database.

Not applicable to models based on SAP BW data sources.

Content model

(dataSourceRef) then (canonicalName then type then procParameters)

Parent elements

definition

suppression

Specifies the behavior of the suppression in the studios (setting the default mode of their sparsity removal capability).

Content model

Content type is NMTOKEN.

The possible values of this element are restricted to the following.

none

Specifies no sort.

null

The studio will suppress empty cells only.

zero

Specifies that this object has no rows.

nullAndZero

The studio will suppress both zero and empty cells.

Parent elements

dataSource

syntaxTip

Describes the syntax that is required by a function.

Attributes

locale

Specifies the locale that applies to the parent object.

Usage: `required` Type: `language`

Content model

Content type is string.

Parent elements

function

table

Defines a database table.

The text may contain macro substitutions. The syntax is defined in the *IBM Cognos Framework Manager User Guide*.

Not applicable to models based on SAP BW data sources.

Content model

Mixed content.

Parent elements

sql

tableType

Defines the type of the underlying table on which the query subject is based. The type can be a table, a view, or unknown. Used by the query engine for fetching a blob.

Content model

Content type is NMTOKEN.

The possible values of this element are restricted to the following.

table

Specifies the table as a table.

view

Specifies the table as a view.

nonTransparentTable

Specifies the table as non-transparent.

unknown

Specifies that the table type is not known.

Parent elements

dbQuery

targetType

The element tag name (type) of the object that is referenced by `refobj`. The shortcut acts as a namespace for object naming purposes only if `targetType` is namespace, folder, or function.

Content model

Content type is NMTOKEN.

Parent elements

shortcut

transactionAccessMode

Specifies the access mode of a new transaction.

Content model

Content type is NMTOKEN.

The possible values of this element are restricted to the following.

unspecified

Specifies that a new transaction will be started in the default mode. In Classic Query Mode, the default mode is read-only.

read

Specifies that a new transaction will be started in read-only mode.

write

Specifies that a new transaction will be started in read-write mode.

Parent elements

dataSource

transactionStatementMode

Specifies the action to be taken when a transaction ends.

Content model

Content type is NMTOKEN.

The possible values of this element are restricted to the following.

unspecified

Specifies that the default action will be taken when a transaction ends. In Classic Query Mode, the default action is rollback. If the underlying database supports autocommit, then the default action will be commit.

rollback

Specifies that a transaction will be rolled back when it ends.

commit

Specifies that a transaction will be committed when it ends.

autocommit

Specifies that autocommit will be carried out for a transaction.

Parent elements

dataSource

treatAs

Specifies the CQE behavior for the shortcut alias. The shortcut will be treated as an alias table in the generated SQL reference. (default) The shortcut will be treated as a simple reference to the target object.

Content model

Content type is NMTOKEN.

The possible values of this element are restricted to the following.

reference

(Default) Shortcut will be treated as a simple reference to the target object.

alias

Shortcut will be treated as an alias table in the generated SQL.

Parent elements

shortcut

type

Specifies the type for a given parent element.

Content model

queryType *(optional)* then interface *(optional)* then functionSetID *(optional)*

Parent elements

dataSource

type

Specifies the type for a given parent element.

Content model

Content type is NMTOKEN.

The possible values of this element are restricted to the following.

unknown

Specifies that the type is not known.

procedure

Specifies the type as procedure.

other

Specifies the type as other than a procedure.

Parent elements

storedProcedure

type

Indicates the type of dimension as either a measure, time, regular, or currency.

Content model

Content type is NMTOKEN.

The possible values of this element are restricted to the following.

regular

Specifies that the dimension is a regular dimension. A regular dimension contains a hierarchy of levels or represents a parent/child hierarchy of members.

time

Defines the dimension as a time dimension.

measure

Specifies that the dimension is a measure or fact dimension. A measure dimension contains measures. It may also contain query items representing the foreign keys of the dimension connected to the measure dimension. In the case of a model measure dimension, query items are not required.

currency

Specifies that the dimension is a currency dimension. Currency dimensions are only valid for OLAP data sources.

Parent elements

dimension

unique

If `true`, the indexed values are unique.

Not applicable to models based on SAP BW data sources.

Content model

Content type is boolean.

Parent elements

index

unSortable

If `true`, data values for this object cannot be compared, and reports cannot sort or group by it. Used mainly for data of type blob.

Content model

Content type is boolean.

Parent elements

calculation , measure , queryItem

updateSubject

Represents an update action for a physical database

Attributes

status

Specifies the status of the execution of a query.

Usage: `optional` Default: `valid` Type: `NMTOKEN`.

Attribute values are restricted to the following.

valid

Specifies that the syntax of the object is valid.

invalid

Specifies that the syntax of the object is not valid.

needsReevaluation

Specifies that the object syntax needs to be reviewed.

Content model

(name *(one or more)* then description *(any number)* then (guid *(optional)* then lastChanged *(optional)* then lastChangedBy *(optional)* then comment *(optional)*) then screenTip *(any number)* then property *(any number)*) then ((definition then (previewFilters *(optional)* then securityFilters *(optional)*) then externalizeMethod *(optional)* then externalizeAutoSummary *(optional)* then useInJoinPath *(optional)* then orderOfMagnitude *(optional)* then querySubjectUsage *(optional)* then useLocalCache *(optional)* then (numberOfRows *(optional)*) *(optional)*) *(optional)*)

Parent elements

folder , namespace

usage

Defines the intended usage of the query item.

Content model

Content type is NMTOKEN.

The possible values of this element are restricted to the following.

unknown

Specifies that the usage is not known.

identifier

Specifies that the object is to be treated as an identifier.

fact

Specifies that the object is to be treated as a fact

attribute

Specifies that the object is to be treated as an attribute.

Parent elements

calculation , measure , queryItem

useInJoinPath

When set to `true`, the parent element will be used when computing the join path between 2 elements.

The governor `contextSensitiveJoinSelection` must be set to `explicit` for `querySubject`, and `shortcut` tagged with `useInJoinPath` to be included in the join path.

The default value is `false`.

Content model

Content type is boolean.

Parent elements

dimension , querySubject , shortcut , updateSubject

useLocalCache

Specifies the use of the **Use Local Cache** query property.

Content model

Content type is boolean.

Parent elements

dimension , querySubject , updateSubject

useV5DataServer

When set to `true`, indicates that dynamic query mode is used.When set to `false`, dynamic query mode is not used. The default is `false`.

Content model

Content type is boolean.

Parent elements

package

value

The value that gets returned when the parameter map locates the key value.

Content model

Content type is string.

Parent elements

parameterMapEntry

value

The value to be passed when the procedure is called.

Content model

Content type is string.

Parent elements

procParameter

valueRef

When a `parameterMap` has been defined, and the unique identifier (`keyRef`) has been found by the query, this value gets returned when the query executes.

Content model

Empty element.

Parent elements

queryItemMap

viewref

References a security view.

Content model

Content type is string.

Parent elements

definition , set

Chapter 4. Metadata Provider (Wrapper) reference

This section contains information about the elements and attributes that are used to package SOAP requests. Each action that passes through the BI Bus API is wrapped with a SOAP envelope. The `mdp.xsd` schema validates each SOAP request.

For each element, the following information is provided:

- A description of the element
- Descriptions of required and optional attributes
- The child elements that the element can or must have
- The parent elements that can contain the element

If you do not specify the value for an optional attribute that has a default value, Framework Manager automatically applies the attribute and uses the default value. We recommend that you review the attributes for the elements you want to use, and set the attribute values if you do not want to use the defaults.

For elements that contain multiple attributes, you may specify the attributes in any order within the element.

action

Contains an individual task to be executed by the underlying metadata provider service, that will either succeed or fail.

The action element has two attributes. The seq attribute is used to specify the order that the actions are performed. The type attribute is used to specify which action is requested.

Content model

Empty element.

Parent elements

transaction

Fault

Contains information regarding any condition other than success resulting from the request, as returned by the metadata provider service. This information is converted to a SOAP fault and returned to the client.

Content model

Empty element.

Parent elements

response

mdprovider

Action request. Defines a request sent to the metadata provider service. Represents an IBM Cognos Framework Manager API (updateMetadata) request that executes an action or series of actions against the model.

Attributes

action

Specifies the action for this request. For an updateMetadata request, the action is always set to execute.

Usage: `required` Type: `NMTOKEN`

model

Specifies the model against which to execute the request.

Usage: `required` Type: `string`.

type

Specifies the type of the request. For an action request, the type is always set to action.

Usage: `required` Type: `NMTOKEN`

Content model

transaction

Parent elements

request

mdprovider

Generic request. Defines a request sent to the metadata provider service. Represents a generic request whose purpose can be to: open or close a client session, or create and save a model. Note: Only single file models and project-based models can be created and saved with a generic request.

Attributes

action

Specifies the required action to be executed by the request.

Usage: `required` Type: `NMTOKEN`.

Attribute values are restricted to the following.

openModel

Specifies a request to open the model.

closeModel

Specifies a request to close the model.

saveModel

Specifies a request to save the model.

createModel

Specifies a request to create the model.

model

Specifies the model against which to execute the request.

Usage: `required` Type: `string`.

type

Specifies the type of the request. For a generic request, the type is always set to generic.

Usage: `required` Type: `NMTOKEN`

Content model

Empty element.

Parent elements

request

request

The IBM Cognos Software Development Kit wrapper element defining the contents as a metadata provider request (mdprovider). There are two metadata provider request types: an action request, and a generic request.

Content model

mdprovider or mdprovider

response

The IBM Cognos Software Development Kit wrapper element that defines the contents of the element as the response from the metadata provider request.

Content model

Fault or response

response

Defines the response to a generic metadata provider service request. It contains only the status of the submitted request.

Content model

status then any

Parent elements

response

status

Defines the status of the submitted generic metadata provider service request.

Attributes

success

Defines the execution of the submitted request as successful.

Usage: `required` Type: `boolean`

Content model

Empty element.

Parent elements

response

transaction

Container element that groups an action request or series of action requests (updateMetadata) submitted to the metadata provider service to execute against a model.

Attributes

commit

An optional attribute that, if specified and set to y, causes the model to be saved after the transaction executes successfully.

Usage: `optional` Type: `NMTOKEN`

seq

Specifies a numeric sequence for the transaction.

Usage: `required` Type: `positiveInteger`

Content model

action

Parent elements

mdprovider

Chapter 5. Custom properties for SAP BW

This section contains information about the custom properties that apply to models built on SAP BW data sources.

For each listed object, the custom properties that apply are described.

folder

If a folder is used to represent an SAP BW dimension (as visible in BEx) or characteristic, folder includes the uniqueName property.

uniqueName

Contains the name of the dimension or characteristic.

Required (for characteristics) or optional (for dimensions)

Property type is hidden.

Information about SAP BW reporting dimensions is not available from the MDDataProviderBW BAPI. Information about SAP BW characteristics is available from the MDDataProviderBW::GetDimensions BAPI.

hierarchy

hierarchyType

By default, Framework Manager assigns hidden as the type for this property. If you wish to provide control over the value of the property, assign string as the type.

The hierarchyType property describes the internal structure of a hierarchy. The values are listed and described in the following table.

Table 7. hierarchyType property values

Value	Usage
default	For query subjects representing a characteristic and not a presentation hierarchy.
characteristic	For query subjects representing a presentation hierarchy in which each non-leaf level is populated with characteristic values from other characteristics
text	For query subjects representing a presentation hierarchy in which each non-leaf level is populated with text values
recursive	For query subjects representing a presentation hierarchy in which all levels are populated with values from a single characteristic.
time	For all query subjects based on any of the date/time SAP BW characteristics, for example, 0CALDAY, 0CALMONTH

Framework Manager and IBM Cognos software do not support presentation hierarchies that use two or more formats for hierarchy construction. For example, where the first three levels are recursive, followed by two levels of characteristic-based values.

Mandatory.

Property type is hidden.

level

dimensionString

For a level in a query subject with a hierarchyType of characteristic, this property contains the name of the characteristic upon which the level is based. The value does not contain any braces, for example, the format is MYDIMENSION, not [MYDIMENSION].

The root level of a presentation hierarchy that contains a not assigned member or node is a special case. In this case, the value of the dimensionString property is HIER_NODE.

dateFormatLength

For each level in a query subject, this property has a type of number, and a value of zero, except for levels associated with any of the date/time characteristics, for example, 0CALDAY, 0CALMONTH. For levels associated with the date/time characteristic, the value of dateFormatLength represents the number of digits contained in the level member unique name that represents date/time.

For example, in the leaf level of the 0CALDAY characteristic, the value of dateFormatLength is 8, as the date appears in the format YYYYMMDD. At the quarter level in one of the 0CALDAY presentation hierarchies, the value of the property is 5, as the quarter is formatted as YYYYQ.

Mandatory.

Property type is number.

dataSource

SAP BW variables

SAP BW variables are parameters of an SAP BW Query that are set up during query definition. When you run the query, the SAP BW variables are filled with values. They function as placeholders and can be processed in different ways. They are automatically exposed as prompts at run time.

SAP BW variable information is included in a composite custom property named **SAP BW Variables** that exists only if a data source has one or more variables associated with it. The **SAP BW Variables** property contains one or more composite properties, each of which must be assigned a unique name. Each property represents a description of a single SAP BW variable. Because the variable information is specified in a custom property, Framework Manager does not validate these properties.

The SAP BW variable information is obtained using the SAP BW BAPI **MDDataProviderBW::GetVariables**.

Framework Manager supports these types of SAP BW variables:

- Characteristic

  There are two kinds of characteristic variables: characteristic value and hierarchy node. Characteristic values variables select characteristic values. Hierarchy node variables select values from any position in a presentation hierarchy.
- Hierarchy

  The user is not prompted for a value because IBM Cognos software automatically populates it at run time, based on the selected hierarchy. Variables for hierarchies function act as placeholders for the hierarchy of a characteristic. All the values for hierarchy variables are read-only.
- Formula

The user types a numeric value at run time. Use formula variables if a formula component should be entered only when a query is run. For example, you can use a formula variable for a value-added tax rate to process the current rate at run time.

- Authorization

  Authorization variables are like other variables, but IBM Cognos software automatically populates the variable values with the user's credentials. SAP BW uses these credentials to supply the information needed by an SAP BW Query that has security applied to it.

Variables for hierarchies function as placeholders for the hierarchy of a characteristic. All the values for hierarchy variables are read-only.

Name property

This property is a string value.

SAP BW equivalent: VARIABLE_NAME

Restrictions: Read-only.

Caption property

The string value for this property is a composite and is locale-dependent. Represent each locale in the model by a custom property whose value is the locale name. For example, if the locales en-ca and fr-fr exist in the model, define two custom properties named en-ca and fr-fr.

The default value is obtained from SAP BW.

Selection type property

The possible values are value, interval, complex, multiValued.

Table 8. Selection type property equivalents

Value	**SAP BW equivalent**
value	SAP_VAR_SEL_TYPE_VALUE
interval	SAP_VAR_SEL_TYPE_INTERVAL
complex	SAP_VAR_SEL_TYPE_COMPLEX
multiValued	SAP_VAR_SEL_TYPE_COMPLEX

Restrictions: Read-only.

Entry type property

The default value is obtained from SAP BW.

Table 9. Entry type property equivalents

Value	**SAP BW equivalent**
optional	SAP_VAR_INPUT_TYPE_OPTIONAL
mandatory	SAP_VAR_INPUT_TYPE_MANDATORY
mandatoryNotInitial	SAP_VAR_INPUT_TYPE_MANDATORY_NOT_INITIAL

Restrictions: Read-only.

Default low value and default high value properties

Each of these properties specifies a range of values.

The default value is obtained from SAP BW.

Restrictions: The **Default High Value** property is applicable only for variables with a **Selection Type** of **interval**.

Description property

This property is a string value.

SAP BW variable type property

The possible values are numeric, characteristic, hierarchy, or hierarchicalNode.

The default is obtained from SAP BW.

Restrictions: Read-only.

Prompt type property

The default value depends on the variable type. If the value of this property is not one of the predefined values, the value used is hierarchyPickList. The predefined values for the prompt type property are as follows.

Table 10. Prompt type property equivalents

Value	**Restrictions**
typeIn	Required for numeric variables and optional for characteristic values
pickList	Optional for characteristic variables
calendar	Only for characteristic variables based on 0CALDAY
hierarchyPickList	Optional for all presentation hierarchies
notApplicable	Required for hierarchy variables

Use this property to specify the type of prompt.

You can improve the performance of variable prompts that use either a picklist or hierarchical picklist. Use the Level Restriction, Initial Number of Picklist Values, and Use Default Value properties to control the performance of those types of variable prompts.

Changing a picklist or hierarchical picklist prompt to a type-in prompt can dramatically improve performance because it does not require the application server to populate a picklist with values. However, it requires your users to be able to accurately enter characteristic values.

Restrictions: Read-only for some types of variables such as characteristic and formula.

Level restriction property

This property is a numeric value.

The default value is **1**.

Use this property to reduce the number of characteristic values that populate a hierarchical picklist. There is a limited number of levels of a hierarchy from which values are obtained.

If the value is zero (0), which is the default, then characteristic values from all levels of a hierarchy (if applicable to the type of prompt) populate the picklist. Otherwise, the property specifies a colon-separated range of levels from which values are obtained (the root level is zero).

For a ragged hierarchy, you must specify all levels that you may want to use even if some branches do not have that level.

Restrictions: Applicable only for hierarchical node variables with a **Prompt Type** of **hierarchyPickList**.

Use default values property

This property is a boolean property that determines whether the default values are used. If this property is set to **true**, users are not prompted for the associated variable, and the default value is always applied.

Use this property to set the variable to a single value. Users are not prompted for the value of a variable and consequently, the IBM Cognos Analytics server does not populate a picklist with values. However, users can no longer change the value of a variable.

The default value is **false**.

Show key and caption property

To show keys and captions for SAP BW variables, set this property to **true**. This property is applicable only for pick list prompts and hierarchy node prompts.

The default value is **false**.

Initial number of pick list values property

This property specifies the initial number of values used to populate a picklist, hierarchical picklist, or prompt.

The default value is zero (0), which means all.

Hidden properties

These properties of SAP BW variables are hidden in the Framework Manager user interface.

Data Type

This property is read only and hidden.

Table 11. Data type property equivalents

Value	SAP BW Equivalent
xsdString	CHAR
xsdDate	CHAR The VAR_TYPE value is SAP_VAR_TYPE_MEMBER and reference dimension is based on 0CALDAY.
memberUniqueName	CHAR The VAR_TYPE value is SAP_VAR_TYPE_MEMBER.
xsdFloat	FLTP
xsdDatetime	DATS
xsdInt	NUMC, DEC, INT1, INT2 or INT4

Table 11. Data type property equivalents (continued)	
Value	**SAP BW Equivalent**
xsdLong	NUMC, DEC, INT1, INT2 or INT4

The default value is xsdString.

Dimension Property

This property is read only and hidden. It applies only to characteristic variables. The SAP BW equivalent for this string value is REFERENCE_DIMENSION.

Hierarchy Property

This property is read only and hidden. It applies only to characteristic variables. The SAP BW equivalent for this string value is REFERENCE_HIERARCHY.

maxNumValues Property

This property specifies the maximum number of values used to populate a picklist, hierarchical picklist, or prompt. It is a numeric value with a default value of 100.

Appendix A. Troubleshooting

This section provides solutions for problems that you may encounter when using the IBM Cognos Framework Manager API.

For troubleshooting information that is not specific to the Framework Manager API, see the Troubleshooting section of the *IBM Cognos Analytics Administration and Security Guide*. You can also refer to component-specific documents.

A protection fault occurs or incorrect results are returned

If the data types `Int8`, `UInt8`, `Int16`, or `UInt16` are used, the custom report functions may not work properly, causing a protection fault or returning incorrect results.

To avoid these problems, use `Int32` for the return type whenever a function must return an integer value.

Error message appears when running BmtScriptPlayer

You may receive an error message such as the following when processing an action script using BmtScriptPlayer:

Runtime Error!Program: ... \bin\BmtScriptPlayer.exe This application has requested the Runtime to terminate it in an unusual way. Please contact the application's support team for more information.

This problem occurs because you are processing an action script that is 16MB or larger.

To resolve this problem, reduce the size of the action script.

Version 1.0 merge actions fail when played back in version 1.1

Merge actions recorded in a version 1.0 action log will fail during playback in version 1.1 because a previously recorded null handle is deemed an invalid parameter in version 1.1. In addition, the MDProvider does not support automatic action log upgrade.

The solution is to run the BMT Script Player or, in Framework Manager, click the Run Script command from the Project menu.

Notices

This information was developed for products and services offered worldwide.

This material may be available from IBM in other languages. However, you may be required to own a copy of the product or product version in that language in order to access it.

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, Nihonbashi-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.

Depending upon the configurations deployed, this Software Offering may use session and persistent cookies that collect each user's

- name
- user name
- password

for purposes of

- session management
- authentication
- enhanced user usability
- single sign-on configuration
- usage tracking or functional purposes other than session management, authentication, enhanced user usability and single sign-on configuration

These cookies cannot be disabled.

If the configurations deployed for this Software Offering provide you as customer the ability to collect personally identifiable information from end users via cookies and other technologies, you should seek your own legal advice about any laws applicable to such data collection, including any requirements for notice and consent.

For more information about the use of various technologies, including cookies, for these purposes, see IBM's Privacy Policy at https://www.ibm.com/privacy/us/en/.

Index

F

G

H

I

J

K

L

M

N

O

P

Q

R

S

T

U

V