IBM OpenPages GRC on Cloud
Version 7.1.0

Report Author's Guide

Note

Before using this information and the product it supports, read the information in “Notices” on page 261.

Product Information

This document applies to IBM OpenPages GRC on Cloud Version 7.1.0 and may also apply to subsequent releases.

Contents

Introduction

This information provides guidance to advanced report authors who develop reports with the IBM® OpenPages® GRC on Cloud reporting tools, using the Report Studio. This document is not meant to be a replacement for the *IBM Cognos 10 Business Intelligence Report Studio User Guide*, but a supplement that is specific to its implementation within OpenPages GRC on Cloud.

CommandCenter must be installed and configured with OpenPages GRC on Cloud. Report authors must attend the IBM OpenPages Report Author training path and have experience using the Report Studio tool.

Disclaimer

The IBM OpenPages documentation describes certain features and functions which may not be available in OpenPages GRC on Cloud. For example, OpenPages GRC on Cloud does not include the ability to submit or start workflow jobs, Capital Modeling, certain administrative functions, or the following modules: Financial Controls Management, Policy and Compliance Management, IT Governance, and Internal Audit Management.

Finding information

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

Accessibility features

Accessibility features help users who have a physical disability, such as restricted mobility or limited vision, to use information technology products.

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

Forward-looking statements

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

Database tool information

The OpenPages GRC on Cloud supports both the IBM DB2® database and the Oracle Database. Use any SQL tool to query these databases.

For example, you could use the Command Line Processor Plus (CLPPlus) or Optim™ Development Studio to run SQL queries on the IBM DB2 database.

Reporting framework icons

In IBM OpenPages GRC on Cloud, the reporting framework icons provide you with a frame of reference in the user interface.

The following table lists the icons that are used within the reporting framework model.

Table 1. Reporting framework icons

Convention	Definition
	Represents a root namespace or any other namespace in a project.
▥	Represents a query subject.
	Represents a query item or in a dimensional model, an attribute.
	Represents a dimension
	Represents a hierarchy.
	Represents a level in a hierarchy.
	Represents a measure.
📁	Represents a folder.
+÷ ×−	Represents a calculation.
	Represents a filter.

IBM Cognos 10 Components and Information

Additional resources that may be helpful to report authors include the online help available from each component's user interface, and the IBM Cognos® 10 Business Intelligence documentation in IBM Knowledge Center (http://www.ibm.com/support/knowledgecenter/SSEP7J/welcome).

The IBM Cognos 10 Business Intelligence components available include:

- IBM Cognos Report Studio is a report authoring tool that professional report authors and developers use to create sophisticated and managed reports.
- IBM Cognos Analysis Studio is used by managers and analysts to better understand their business and to get answers to questions that they have about their business.
- IBM Cognos Query Studio is the reporting tool for creating simple queries and reports in IBM Cognos Business Intelligence.
- IBM Cognos Workspace is a dashboard authoring tool that you use to build and use sophisticated, interactive workspaces using IBM Cognos content, as well as external data sources.
- IBM Cognos Workspace Advanced is a report authoring tool that you use to author simple reports and explore your data.

- IBM Cognos Connection is the Web portal for IBM Cognos Business Intelligence. It is the starting point to access your BI information and the functionality of IBM Cognos BI.
- IBM Cognos Framework Manager is a metadata modeling tool that drives query generation for IBM Cognos software.

Chapter 1. Database Models

The database model topics give the report author the background information that is required to understand the process for creating reports using the Report Studio tool. This includes information about the structure of the application data model of the IBM OpenPages GRC on Cloud software, and the corresponding reporting database tables.

The Application Data Model

The application data model of IBM OpenPages GRC on Cloud is designed using object oriented techniques for the purpose of maintainability, cost and extensibility.

The OpenPages GRC on Cloud object model is highly configurable because it consists of object types, properties of those objects and the allowable relationships between these object types.

The core of the database design consists of the following tables:

- RESOURCE - This table contains an instance of each object created in the system.
- RESRELATIONSHIPS - This table contains all relationships between objects in the system.
- ASSETTYPES table - The definition of the allowable content type (object types) in the system.
- ASSETTYPESBUNDLEDEFS - Contains the relationships between Asset Types and BundleDefs.
- BUNDLEDEFS - A field group attached to an object.
- PROPERTYDEFS - The fields attached to an instance of an object.
- PROPERTYVALS - The field values attached to an instance of an object.

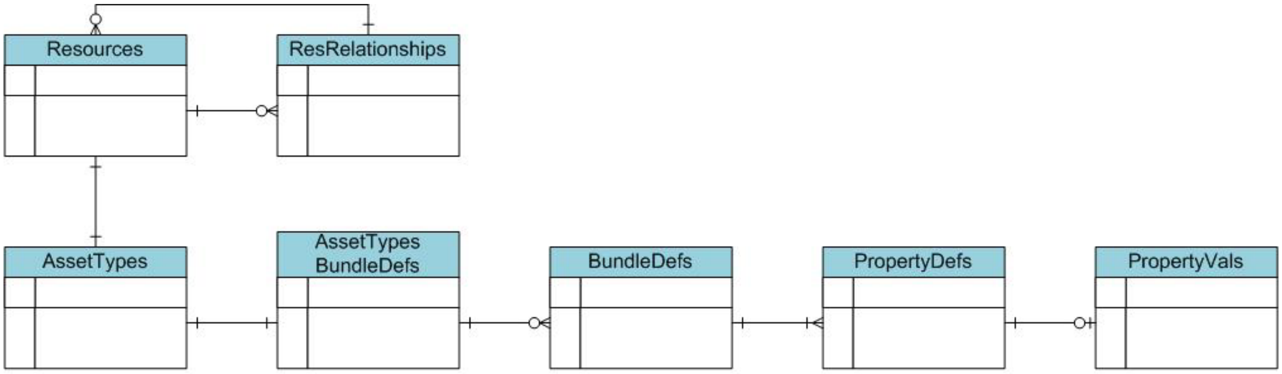


Figure 1. Core Database ER Diagram

Through these generic defined tables, any type of object can be created in the OpenPages GRC on Cloud database. This gives our customers the ability to extend existing system objects or add new objects with their own definitions.

There are many other tables used in support of these tables, but these comprise the fundamental tables required to create objects in OpenPages GRC on Cloud.

The advantages of this database design can be seen in that a small set of Java™ classes can create and access any object in the database. The disadvantage to this design is that no commercial reporting tool can be used to generate reports from this design due to the following:

- The relationship between the RESOURCE and RESRELATIONSHIP table is recursive.
- In order to create a report on an object, many tables need to be joined. Eventually this exceeds the limits of a reporting tool.
- To report from these tables often requires the database designers knowledge of how they were constructed.

Note: You should not write reports against the core database as it will affect your system performance.

Reporting Table Design

To facilitate reporting from the application data database, a secondary set of views is created that combines the various individual tables from the application data design into a format that is better suited for generating reports.

Table 2 lists the naming convention for these tables and views.

Where `<OBJECT_TYPE>` is the name of the type of object, and `<ENUMERATION>` is the name of a multivalued enumerated string field.

Table 2. Reporting Table Naming Convention

This naming convention...	Indicates this is a...
`RV_<OBJECT_TYPE>` (RV_ with a single underscore)	Object view For example, `RV_ENTITY`
`RT_<OBJECT_TYPE>_<ENUMERATION>` (RT_ with a single underscore)	Multivalued enumeration table For example, `RT_RISKASSESSMENT_DOMAIN`
`RT__<OBJECT_TYPE>_<OBJECT TYPE>` (RT__ with a double underscore)	Relationship table between 2 object types For example, `RT__ENTITY_PROCESS`

Important:

- Oracle has a limitation of 30 characters for a table name. Some table names will be truncated.
- The RV views are different from the RT tables in that the views have data security built into them while the RT tables do not.

Most objects that are defined in IBM OpenPages GRC on Cloud will have both an RV_<OBJECT_TYPE> view and an RT_<OBJECT_TYPE> table in the database.

However, joining tables do not have views because they do not need security. When using direct SQL, it is preferable to use the RV views due to the data security code being already present.

The relationship tables represent the parent-child relationships that are defined during the installation of your system.

The enumeration tables store the relationship of objects to enumerated strings in proper relational form.

These tables are automatically generated based upon settings under the Administration menu in OpenPages GRC on Cloud. More information on this topic can be found in the *IBM OpenPages GRC on Cloud Administrator's Guide*.

Object Tables

This topic lists all of the system fields that will be included in all the object tables. All other fields are customer specific and correspond to the particular configuration of your system; they are not shown here.

Table 3. Object Table System Fields

Column Name	Description	Native Type
CHECKED_IN_BY	ID of the user who last modified this object.	NUMBER
CREATION_DATE	Object creation date.	DATE
CREATOR	Object creator ID.	NUMBER
DESCRIPTION	Object description.	VARCHAR2(2048)
DETAIL_PAGE_URL	The URL of system page for this object.	VARCHAR2(2083)
FULL_PATH	The object full path.	VARCHAR2(1024)
IS_LOCKED	Object lock status. Y or N.	NUMBER
LATEST_VERSION	Latest object version.	NUMBER
MAJOR_NUMBER	Major version number.	NUMBER
MINOR_NUMBER	Minor version number.	NUMBER
MODIFICATION_DATE	Last modification date of the object.	DATE
NAME00	Object name.	VARCHAR2(256)
PARENT_FOLDER	Object parent folder. Resource ID.	NUMBER
REPORTING_PERIOD_ID	The ID of the user specified reporting period label.	NUMBER
RESOURCE_ID	Unique resource ID of the object.	NUMBER
VERSION_COMMENTS	Version comments.	VARCHAR2(2048)

Table 4 lists the indexes that are located on all object tables:

Table 4. Object Table Indexes

Index	Column Names
1	PARENT_FOLDER, REPORTING_PERIOD_ID

Table 4. Object Table Indexes (continued)

Index	Column Names
2	FULL_PATH, REPORTING_PERIOD_ID
N	<ENUMERATION> Fields (one index for each field)
N	Currency Fields (one index for base currency and one index for local currency)
System	ENTITY_ID, REPORTING_PERIOD_ID

Object Views

There is a corresponding view for each object defined in the system. The views differ from the object tables in that they have data security built into their definition.

Table 5 lists 3 additional fields included with each view:

Table 5. Additional Fields Included with View

Column Name	Description	Native Type
OP_OBJECT_TYPE_ID	The type of object in the view.	NUMBER
OP_RECURSIVE_NATIVE_LEVEL	In a hierarchical object, the level of the hierarchy.	NUMBER
OP_FLAG_READ_ACCESS	A "Y" or "N" flag indicating whether the user has the privilege to view the data.	VARCHAR2(4)

Enumeration Tables

This class of tables contains lists of enumerated variables that apply to a particular object type where the multi-select enumerated values are stored as separate records.

All enumeration tables contain the same fields, listed in Table 6.

Table 6. Enumerated Table Fields

Column Name	Description	Native Type
<PARENT>_ID	The unique ID of the parent object.	NUMBER
REPORTING_PERIOD_ID	The ID of the user specified reporting period label.	NUMBER
<ENUMERATION>	The ID of the enumerated string.	NUMBER

Table 7 lists the indices that are located on all enumeration tables:

Table 7. Enumeration Table Indices

Index	Column Names
1	RESOURCE_ID, REPORTING_PERIOD_ID
2	<ENUMERATION>

Table 7. Enumeration Table Indices (continued)

Index	Column Names
System	RESOURCE_ID, REPORTING_PERIOD_ID, <ENUMERATION>

The Relationship Table

Relationship tables represent the many-to-many relationships between IBM OpenPages GRC on Cloud objects.

All relationship tables consist of the same fields, listed in Table 8.

Table 8. Relationship Table Fields

Column Name	Description	Native Type
PARENT_ID	The unique ID of the parent object.	NUMBER
REPORTING_PERIOD_ID	The ID of the user specified reporting period label.	NUMBER
CHILD_ID	The unique ID of the child object.	NUMBER
HIER_INSTANCE_ID	This value is always set to "1".	NUMBER
IS_PRIMARY	Indicates whether the parent object is the primary parent of the child object. The values are "Y" or "N".	VARCHAR2(1)
DISTANCE	The distance between parent and child object. If the distance between the parent and child objects is "1", the objects are linked directly.	NUMBER

Table 9 lists the indices that are located on all relationship tables:

Table 9. Relationship Table Indices

Index	Column Names
1	PARENT_ID, REPORTING_PERIOD_ID
2	CHILD_ID, REPORTING_PERIOD_ID
System	PARENT_ID, CHILD_ID, REPORTING_PERIOD_ID, HIER_INSTANCE_ID

Chapter 2. Framework Data Models

The framework data model has two branches: a relational namespace and a dimensional namespace.

The dimensional namespace has been introduced specifically to give IBM OpenPages GRC on Cloud users access to IBM Cognos Analysis Studio and the OLAP features that are available.

The Relational Data Model

The relational data model that IBM OpenPages GRC on Cloud provides is designed to enable the user to report on objects based upon their defined relationships in the system.

The data is stored in a "normalized" set of tables that provides efficient storage. The goal is to avoid data redundancy and provide a high level of performance. The organization of this type of data typically takes a skilled business analyst or data modeler to understand and is not designed to be seen by an end user.

This type of model is typically used in list reports where mixed data (numeric, date, string) is required to convey information to the users. Aggregation can be performed in this type of report but it is not a key element of the report.

The key characteristics of the relational data model are as follows:

- Modeled as an Entity-Relationship diagram.
- Highly normalized data.
- Typically divided up by objects related to other objects.
- All attributes for an object including textual as well as numeric, belong to the object.

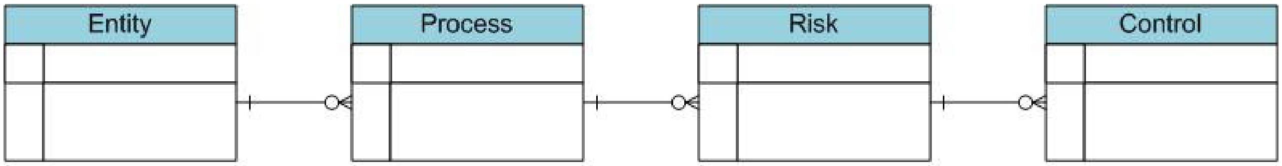


Figure 2. Entity-Relationship Diagram

The above diagram depicts the typical entity-relationship diagram of four of the major objects in OpenPages GRC on Cloud.

Relational Modeling Terminology

This topic lists relational modeling terminology.

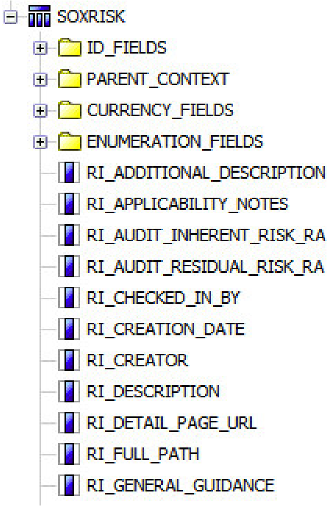
Table 10. Relational Modeling Terminology

Icon	Name	Description
▥	Query Subject	Is a SQL query that has been designed to expose a certain set of data to the user for reporting purposes.
▮	Query Item	A database field.

Folder View of a Relational Data Model

This topic shows a folder view of a relational structure.

Figure 3 shows the SOXRisk query subject and some of its query items.



SOXRISK
ID_FIELDS
PARENT_CONTEXT
CURRENCY_FIELDS
ENUMERATION_FIELDS
RI_ADDITIONAL_DESCRIPTION
RI_APPLICABILITY_NOTES
RI_AUDIT_INHERENT_RISK_RA
RI_AUDIT_RESIDUAL_RISK_RA
RI_CHECKED_IN_BY
RI_CREATION_DATE
RI_CREATOR
RI_DESCRIPTION
RI_DETAIL_PAGE_URL
RI_FULL_PATH
RI_GENERAL_GUIDANCE

Figure 3. Relational Folder View

There will be one query subject for each type of object defined in your system. Each of these objects will be related by left outer joins to one another based upon the parent-child relationships defined in your system. Consult the *IBM OpenPages GRC on Cloud Administrator's Guide* for more information on system configuration.

The Dimensional Data Model

A dimensional data model is designed to address the needs of end users by providing a data structure that is designed in business terminology. The data is grouped around facts, which are numerical fields that can be aggregated and analyzed, and dimensions which are business filters and grouping fields. This data is very well suited for cross-tab and graphical reports.

This type of model is often referred to as multi-dimensional, OLAP or hypercubes. The tools provided by IBM OpenPages GRC on Cloud enables you to define the facts and dimensions required to meet your reporting needs.

The key characteristics of the dimensional data model are as follows:

- Data is modeled as dimensions and facts.

- Dimensions can be considered "Filters" or "Group By" expressions.
- Facts are numeric data.
- Data is modeled as a Star or Snowflake schema.

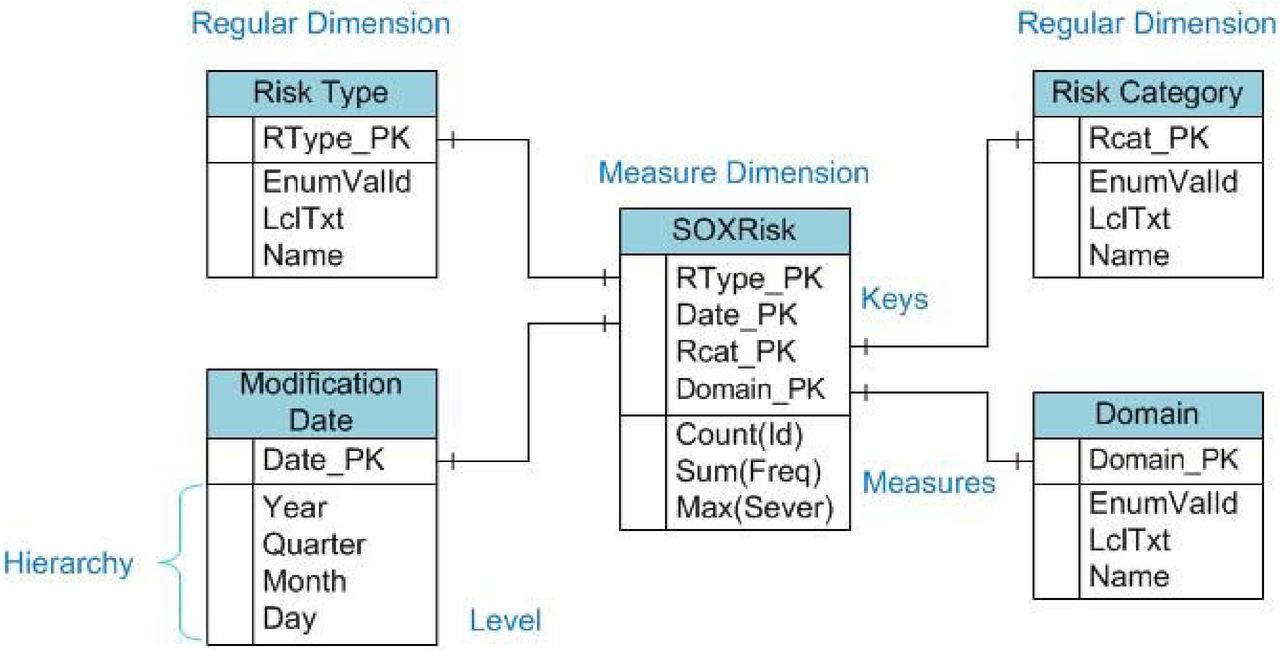


Figure 4. Dimensional Data Model

The above figure details a SOXRisk Measure Dimension that contains the key facts for the Risk object. Joining to it are the Regular Dimensions of Risk Type, Modification Date, Risk Category and Domain. By this design, the data in the Measure Dimension can be filtered by any of the dimensions.

Dimensional Modeling Terminology

This topic lists dimensional modeling terminology.

Table 11. Dimensional Modeling Terminology

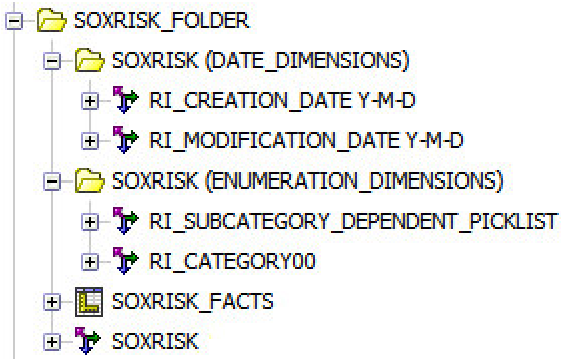
Icon	Name	Description
	Regular Dimension	Is a grouping of descriptive data about an aspect of a business such as risk categories, occurrence dates, or domain. These are typically string values. All regular dimensions will include a hierarchy as part of their definition.
	Hierarchy	The hierarchy is a series of parent-child relationships, typically where a parent member represents the consolidation of the child members. Hierarchies can include single levels where parents have no children.
	Level	A member of a hierarchy that represents a business requirement for consolidation of data.
	Attribute	Is a query item that is a data member of a level that can be added to a report.
	Measure Dimension	Is a collection of measures.

Table 11. Dimensional Modeling Terminology (continued)

Icon	Name	Description
📐	Measures	Is any numeric query item, which can undergo mathematical aggregation and analysis.

Folder View of a Dimensional Data Model

This topic shows a folder view of a dimensional structure.



- SOXRISK_FOLDER
  - SOXRISK (DATE_DIMENSIONS)
    - RI_CREATION_DATE Y-M-D
    - RI_MODIFICATION_DATE Y-M-D
  - SOXRISK (ENUMERATION_DIMENSIONS)
    - RI_SUBCATEGORY_DEPENDENT_PICKLIST
    - RI_CATEGORY00
  - SOXRISK_FACTS
  - SOXRISK

Figure 5. Dimensional Model Structure

By creating a dimensional model, the following features are now available to the report:

- Drill Down/Up - Drilling down or up is a specific analytical technique whereby the user navigates among levels of data ranging from the most summarized (up) to the most detailed (down).
- Roll-up - A roll-up involves computing all of the data relationships for one or more dimensions.
- Slice - A slice is a subset of the data corresponding to a single value for one or more members of the dimensions not in the subset.

Chapter 3. Object Model

The object model referenced throughout this document is the standard (out-of-the-box) model that is installed for all modules in a first-time installation.

Only objects within the default object model that are common to all modules are included and discussed in this document for simplicity.

Important: All examples in this document are displayed in Report Design Language (RDL).

Object Model Terminology

This topic lists basic terms needed to understand the IBM OpenPages GRC on Cloud object model.

Table 12. Object Model Terminology

Term	Description
Multiple Parents	A child object that is associated to more than one parent.
Parent-Child Relationship	When an object becomes associated to another object, the two objects are considered to be a 'Parent' object and a 'Child' object. The relationship that is formed is a 'Parent-Child' relationship.
Primary Association	A parent can have multiple child objects and all of them can be primary. A child can have multiple parent objects and only one of them can be primary. Primary associations are used to determine the path the system should follow when executing a number of operations that require object hierarchy traversal.
Primary Object	A core object of OpenPages GRC on Cloud.
Recursive Relationship	A relationship in which a child object is the same object type as the parent.
Secondary Object	An object that can be associated to multiple object types in the common spine.

Object Model Characteristics

This topic describes the characteristics of the standard (out-of-the-box) object model.

The characteristics are:

- The business entity and sub-process object types are recursive.
- The Issue, Signature, File, and Link object types can be associated to any primary object type.
- Object types can contain single valued and multi-valued enumerated string data types.

Note: Depending on your configuration, the object model that you see may be different.

Table 13 lists the primary object types that are shared across all modules. The list is followed by Figure 6 showing the object model and relationship among primary objects. All objects have a direct parent-child relationship.

Table 13. Primary Object Types

Object Type	Label
SOXBusEntity (recursive)	Business Entity
SOXProcess	Process
SOXSubprocess (recursive)	Sub-Process
SOXRisk	Risk
SOXControl	Control
SOXTest	Test Plan
SOXTestResult	Test Result
RiskAssessment	Risk Assessment

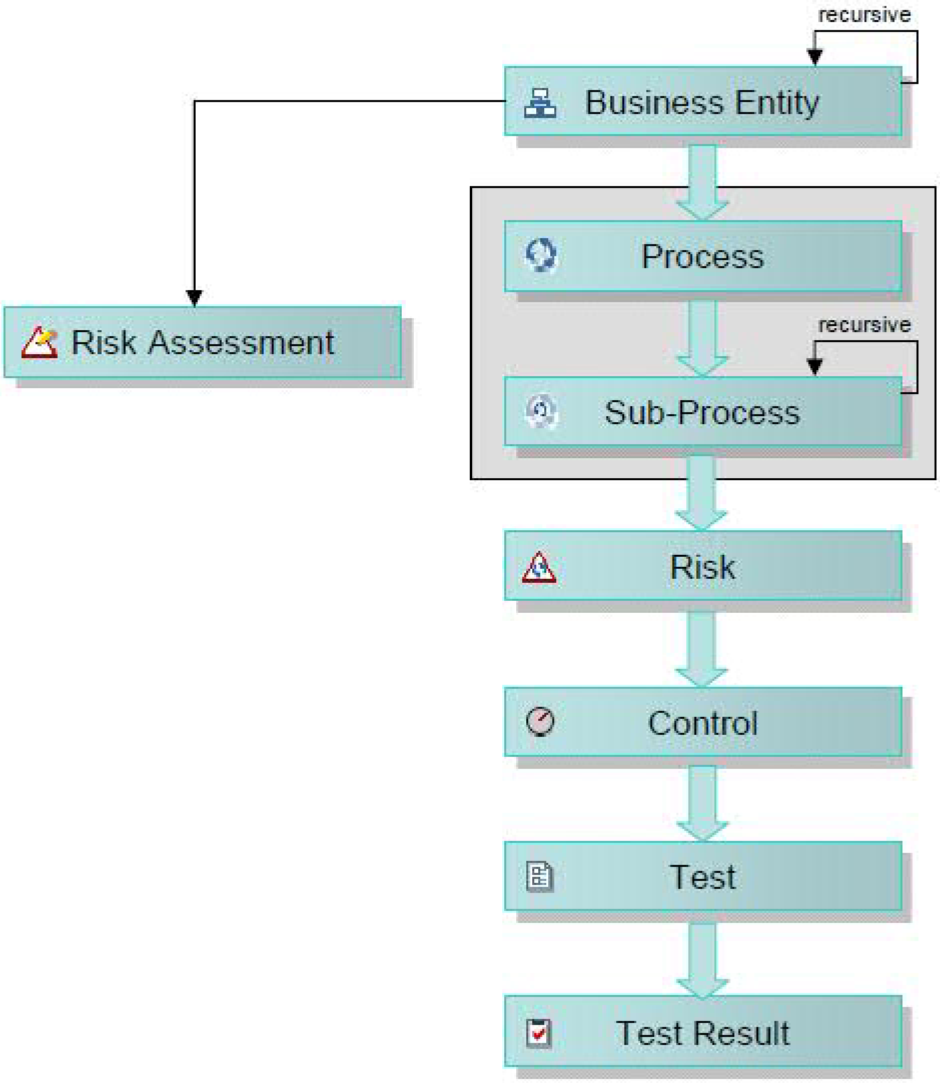


Figure 6. Primary Object Relationships

Table 14 lists the secondary object types that are shared across all modules. The list is followed by Figure 7 showing the secondary objects without links to the primary objects because the joins are formed in the reporting tool.

Table 14. Secondary Object Type

Object Type	Label
SOXIssue	Issue
SOXTask	Action Item
SOXSignature	Signature
SOXExternalDocument	Link
ProjectActionItem	Action Item
SOXDocument	File
Milestone	Milestone

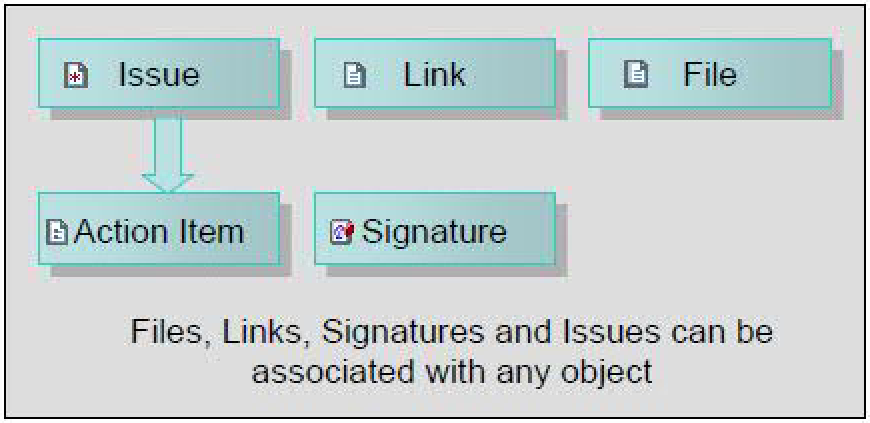

Issue
Link
File
Action Item
Signature
Files, Links, Signatures and Issues can be associated with any object

Figure 7. Secondary Objects

This model represents the basic relationship of objects within IBM OpenPages GRC on Cloud. For the exact relationships for your own system either obtain the object model diagram that was created during the system installation process or examine the object relationships from the **Administrator | Object Types** menu.

Chapter 4. Reporting Framework

The reporting framework is based on a customer’s object model and allowable relationships. The framework generates a set of query subjects and adds the appropriate relationships and query item information to those query subjects.

The reporting framework provides you with the following benefits:

- Supports creating reports based on the IBM OpenPages GRC on Cloud object model hierarchy that you configure.
- Supports recursive object types (such as business entity and sub-process).
- Supports child object types that can have multiple parents.
- Contains built-in security filters that replicate OpenPages GRC on Cloud application security.
- Supports localized values including single value and multivalued enumerated string data types.

Add locales to the reporting framework

You must add locale codes to the IBM OpenPages GRC on Cloud Reporting Framework V6 before regenerating the framework so they can be applied to localized reporting and reports.

Administration > Application > Settings > Platform > Reporting Framework V6 > Configuration > Locales

Default: en_US

Values: In the **Value** field, type the following locale code values:

- en_US (U.S. English)
- de_DE (German)
- en_GB (U.K. English)
- es_ES (Spanish)
- fr_FR (French)
- it_IT (Italian)
- ja_JP (Japanese)
- pt_BR (Brazilian Portuguese)
- zh_CN (Simplified Chinese)
- zh_TW (Traditional Chinese)

The Framework Package

The generated model for the reporting framework is published under the OPENPAGES_REPORTS_V6 package.

Namespaces

A namespace uniquely identifies a collection of query subjects, their relationships, and other objects (such as calculations) that you can use for authoring reports.

The framework generator uses the definition of a namespace (which is defined in the IBM OpenPages GRC on Cloud user interface) to create a corresponding namespace in the framework model. For information about creating or modifying namespaces, see the *IBM OpenPages GRC on Cloud Administrator's Guide*.

The Default Namespace

The `DEFAULT` namespace contains the out-of-the-box query subjects, calculations and query items and should not be changed. It contains a majority of objects and associations present in the IBM OpenPages GRC on Cloud predefined default object hierarchy. The default namespace was designed to meet the most common reporting requirements.

Under the default namespace are two other namespaces: DEFAULT_REL and DEFAULT_DIM. These represent the relational data model and the dimensional data model.

As shown in Figure 8, the generated OpenPages GRC on Cloud framework model contains one default namespace which includes the two sub-namespaces.

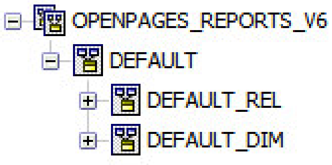


Figure 8. Generated Framework Model Namespaces

In addition to the DEFAULT namespace, there are other namespaces that come pre-defined depending on the solutions installed on your system. For details, see Appendix B, "Namespace Definitions," on page 245.

Naming Conventions

- All query subjects that represent an object defined in IBM OpenPages GRC on Cloud are named the same as the object.
- All query items for an object are prefixed with a unique identifier for that object. The list of identifiers can be found in the *IBM OpenPages GRC on Cloud Administrator's Guide.*
- All calculations that are associated with an object have a name beginning with the object name.
- If object type dimensions have been defined, the query subject name begin with level names defined in the dimension.

The Relational Model Top Level Folders

Note: Some of the workflow functionality described in this topic might not be available in IBM OpenPages GRC on Cloud.

Folders are used extensively to organize query subjects so report authors can find items easily and intuitively. Figure 9 on page 17 shows the top level folder layout for the DEFAULT_REL relational data sub-namespace.

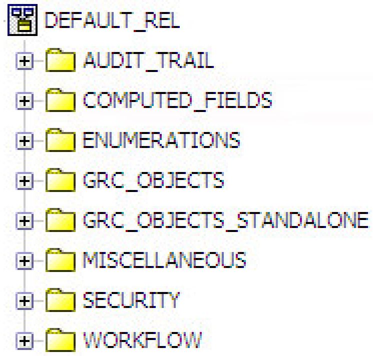


Figure 9. Relational Model Folder Layout

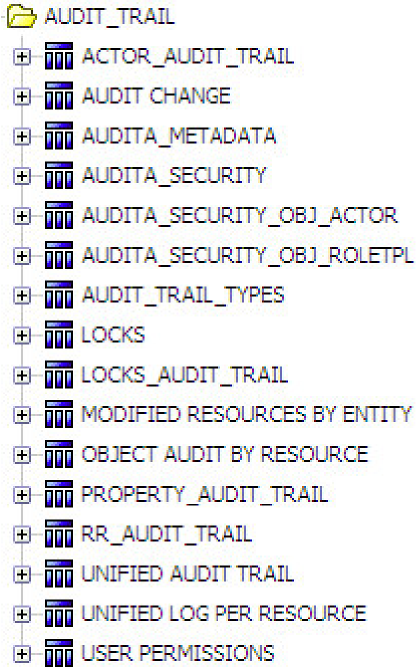
Top-level folders can contain other folders, query subjects, and query items, as listed in Table 15.

Table 15. Top-Level Folder

This Top-Level Folder...	Contains Query Subjects for...
AUDIT_TRAIL	System audit tables
COMPUTED_FIELDS	Computed fields that have been created in the application
ENUMERATIONS	Prompt enumeration query subjects
GRC_OBJECTS	Primary compliance objects
GRC_OBJECTS_STANDALONE	Secondary compliance objects
MISCELLANEOUS	Information not related to any other folder
SECURITY	System security
WORKFLOW	Workflow objects

The Audit Trail Folder (Relational Model)

Figure 10 on page 18 shows the Audit Trail folder with all of its query subjects.



AUDIT_TRAIL
- ACTOR_AUDIT_TRAIL
- AUDIT CHANGE
- AUDITA_METADATA
- AUDITA_SECURITY
- AUDITA_SECURITY_OBJ_ACTOR
- AUDITA_SECURITY_OBJ_ROLETPL
- AUDIT_TRAIL_TYPES
- LOCKS
- LOCKS_AUDIT_TRAIL
- MODIFIED RESOURCES BY ENTITY
- OBJECT AUDIT BY RESOURCE
- PROPERTY_AUDIT_TRAIL
- RR_AUDIT_TRAIL
- UNIFIED AUDIT TRAIL
- UNIFIED LOG PER RESOURCE
- USER PERMISSIONS

Figure 10. Audit Folder

Table 16 is details the query subjects in this folder.

Table 16. Audit Folder Query Subjects

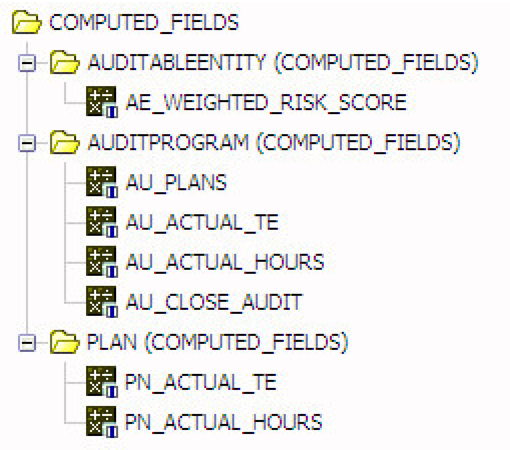
Query Subject	Description
Actor_Audit_Trail	This table contains records related to a user's login attempts.
Audit Change	This query returns a list of properties that have been changed for any object associated with an entity in the system. The following prompts are defined for this query subject: • Start Date • End Date • Entity ID • Set ID (always set = 1)
Audita_MetaData	This view contains audit trail information related to the registry settings.
Audita_Security	This view contains audit trail information related to the creation of user security.
Audita_Security_Obj_RoleTpl	This view returns audit information for role templates, role template assignments, role template application permissions and role template objects Access Control Lists (ACLs).
Audita_Security_Obj_Actor	This view returns audit information for actors and actor ACLs.

Table 16. Audit Folder Query Subjects (continued)

Query Subject	Description
Audit_Trail_Types	This table contains the names and descriptions of all the types of audit trail items that can be recorded.
Locks	This table contains resource lock data.
Locks_Audit_Trail	This table contains the audit trail of all locks that were applied in the system.
Modified Resources by Entity	This query subject returns a list of resources, associated to a particular entity, that have been modified. The following prompts are defined for this query subject: • Start Date • End Date • Parent Entity ID • Set ID (always set = 1)
Object Audit by Resource	This query subject returns a list of all property definitions in the system and whether they have been marked for deletion.
Property_Audit_Trail	This table holds all the records for any property that has been changed in the system.
RR_Audit_Trail	This table contains audit trail data for the resrelationship table. Any modification of relationships is written to this table.
Unified Audit Trail	This query subject returns a list of all object properties in the system that have been changed.
Unified Log Per Resource	This query subject returns a list of all object properties that have been changed for the specified resource ID. The following prompts are defined for this query subject: • Start Date • End Date • Resource ID • Set ID (always set = 1)
User Permissions	This query subject returns a list of all users in the system and their effective rights.

The Computed Fields Folder (Relational Model)

Figure 11 on page 20 shows an example of a computed fields folder and its sub-folders. This folder will only get created if you have a computed field defined in your namespace.



- COMPUTED_FIELDS
  - AUDITABLEENTITY (COMPUTED_FIELDS)
    - AE_WEIGHTED_RISK_SCORE
  - AUDITPROGRAM (COMPUTED_FIELDS)
    - AU_PLANS
    - AU_ACTUAL_TE
    - AU_ACTUAL_HOURS
    - AU_CLOSE_AUDIT
  - PLAN (COMPUTED_FIELDS)
    - PN_ACTUAL_TE
    - PN_ACTUAL_HOURS

Figure 11. Computed Fields Folder

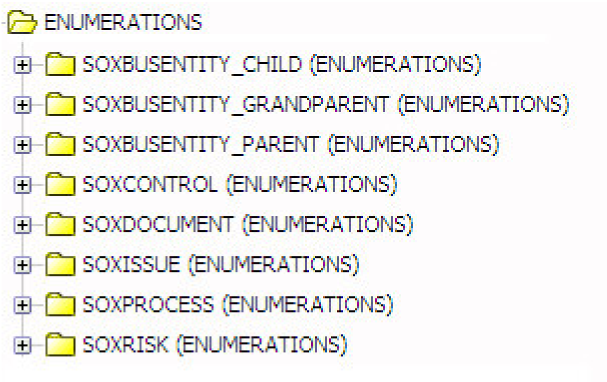
Table 17 is an explanation of the folders in the Computed Fields Folder.

Table 17. Enumeration Folder

Folder	Description
<Object Name> (COMPUTED_FIELDS)	A sub-folder will be created for each object in the namespace that has a computed field.
<Computed Field Name>	Each computed field that has been created will be listed here.

The Enumerations Folder (Relational Model)

Figure 12 shows an example of the Enumerations folder and its sub-folders. This folder holds query subjects which can only be used for creating prompts in reports. There will be a folder for each object type. The objects shown will be different for each user's installation.



- ENUMERATIONS
  - SOXBUSENTITY_CHILD (ENUMERATIONS)
  - SOXBUSENTITY_GRANDPARENT (ENUMERATIONS)
  - SOXBUSENTITY_PARENT (ENUMERATIONS)
  - SOXCONTROL (ENUMERATIONS)
  - SOXDOCUMENT (ENUMERATIONS)
  - SOXISSUE (ENUMERATIONS)
  - SOXPROCESS (ENUMERATIONS)
  - SOXRISK (ENUMERATIONS)

Figure 12. Enumerations Folder

Table 18 on page 21 is an explanation of the folders in the Enumerations Folder.

Table 18. Enumeration Folder

Folder	Description
<Object Name> (ENUMERATIONS)	These folders contain the query subjects corresponding to each enumerated string defined for an object. These should only be used to create prompts in reports.

Enumerated Strings (Relational Model)

Figure 13 shows an example of an opened ENUMERATIONS folders:

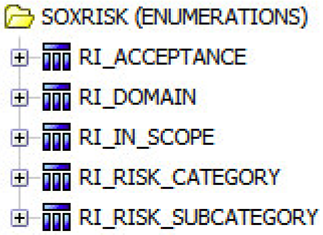

SOXRISK (ENUMERATIONS)
RI_ACCEPTANCE
RI_DOMAIN
RI_IN_SCOPE
RI_RISK_CATEGORY
RI_RISK_SUBCATEGORY

Figure 13. ENUMERATIONS Folder

Each query subject represents an enumerated string that has been defined for that object. These enumerated strings can be either single or multivalued.

Figure 14 shows an example of an opened query subject:

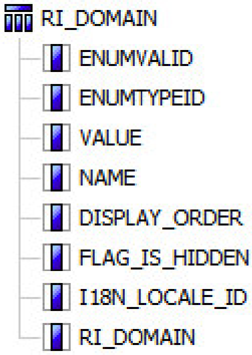

RI_DOMAIN
ENUMVALID
ENUMTYPEID
VALUE
NAME
DISPLAY_ORDER
FLAG_IS_HIDDEN
I18N_LOCALE_ID
RI_DOMAIN

Figure 14. Enumerated String Query Items

The following table lists query items found in an enumerated string:

Table 19. Multivalued Enumerated String

Query Subject	Description
ENUMVALID	This is the ID of the of the enumerated string value.
ENUMTYPEID	The is the ID of the type of variable returned. All records in an enumerated string will return the same value.
VALUE	This is a value returned for each record.
NAME	This is the name of this record in the enumerate string returned in RDL.

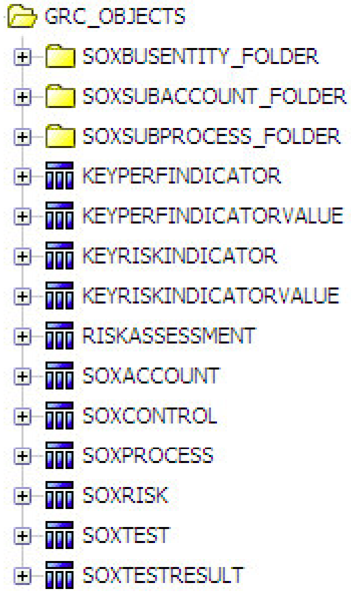
Table 19. Multivalued Enumerated String (continued)

Query Subject	Description
DISPLAY_ORDER	This is a numeric value indicating the order in which the value is displayed to the user.
FLAG_IS_HIDDEN	This is a true/false value indicating whether the value is displayed to the user.
I18N_LOCALE_ID	This is locale of the returned value.
RI_DOMAN	This is the localized name of this record in the enumerated string.

These query subjects can ONLY be used for creating prompts. You would use the ENUMVALID as the returned value and the RI_DOMAIN as the display value, because it is localized. Other parameters can be used for filtering.

The GRC Objects Folder (Relational Model)

Figure 15 shows an example of the GRC Objects folder with all of its sub-folders. The objects shown will be different for each user's installation.



GRC_OBJECTS
SOXBUSENTITY_FOLDER
SOXSUBACCOUNT_FOLDER
SOXSUBPROCESS_FOLDER
KEYPERFINDICATOR
KEYPERFINDICATORVALUE
KEYRISKINDICATOR
KEYRISKINDICATORVALUE
RISKASSESSMENT
SOXACCOUNT
SOXCONTROL
SOXPROCESS
SOXRISK
SOXTEST
SOXTESTRESULT

Figure 15. GRC Objects Folder

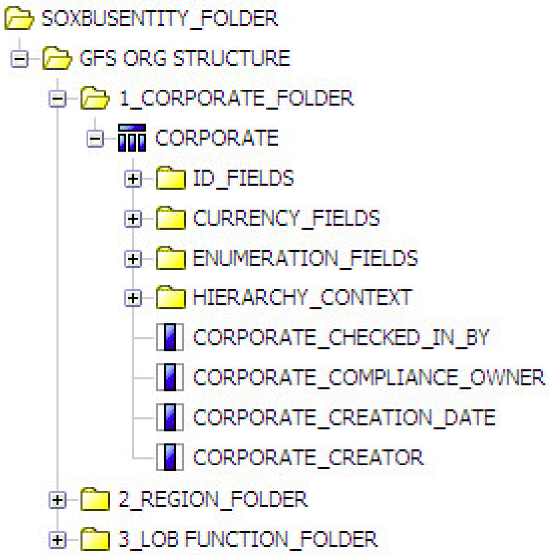
Table 20 lists the folders in the GRC Folder.

Table 20. GRC Objects Folders

Folder	Description
<Recursive Object>_FOLDER	This folder contains all of the query subjects related to the recursive objects used in this namespace.
<Object Name>	These query subjects contain the data items related to the objects specified in this namespace.

The SOXBusEntity Folder (Relational Model)

Figure 16 shows an example of a SOXBusEntity object folder with all of its items.



```
SOXBUSENTITY_FOLDER
  GFS ORG STRUCTURE
    1_CORPORATE_FOLDER
      CORPORATE
        ID_FIELDS
        CURRENCY_FIELDS
        ENUMERATION_FIELDS
        HIERARCHY_CONTEXT
        CORPORATE_CHECKED_IN_BY
        CORPORATE_COMPLIANCE_OWNER
        CORPORATE_CREATION_DATE
        CORPORATE_CREATOR
    2_REGION_FOLDER
    3_LOB FUNCTION_FOLDER
```

Figure 16. SOX Business Entity Object Folder

Table 21 lists the items in SOXBusEntity folder.

Table 21. Entity Object Folder

Folder/Query Subject	Description
<Hierarchy>	A folder will be created for each recursive object level created in IBM OpenPages GRC on Cloud.
<Level>_FOLDER	A folder will be created to hold all query subjects related to a single level of the recursive object.
ID_FIELDS	For each level of the recursive object, this folder holds the ID field query items.
CURRENCY_FIELDS	For each level of the recursive object, this folder holds the currency field query items.
ENUMERATION_FIELDS	For each level of the recursive object, this folder holds the object specific enumerated strings.
HIERARCHY_CONTEXT	For each level of recursive object, this folder holds data items that enable the report author to filter this level of the object.
<Query Items>	For each level of the recursive object, these are all the remaining query items for the level.
SOXBUSENTITY_GPC	This is a single query subject that contains all GPC query items. This is identical to the SOXBUSENTITY query subject in the legacy framework

The GPC Query Subjects (Relational Model)

Figure 17 shows the Grandparent-Parent-Child folder structure underneath the Entity folder.

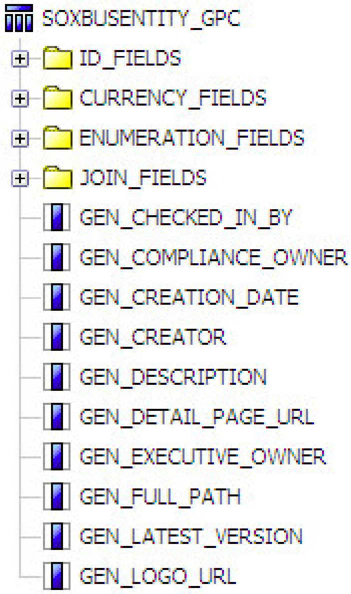


Figure 17. GPC Folder Items

Table 22 lists the items in the GPC folder.

Table 22. GPC Items

Folder/Query Subject	Description
SOXBUSENTITY_GPC	This is the query subject containing all the fields for each of the standard entity levels.
ID_FIELDS	These are all the ID fields that are associated with this entity level.
CURRENCY_FIELDS	These are all the currency fields associated with this entity level.
ENUMERATION_FIELDS	These are all the enumerated string fields associated with this entity level.
JOIN_FIELDS	For the GPC object, this folder holds data items that enable the report author to filter the levels of this object.
<Query Items>	These are all of the query items that can be used in a report for the entity level.

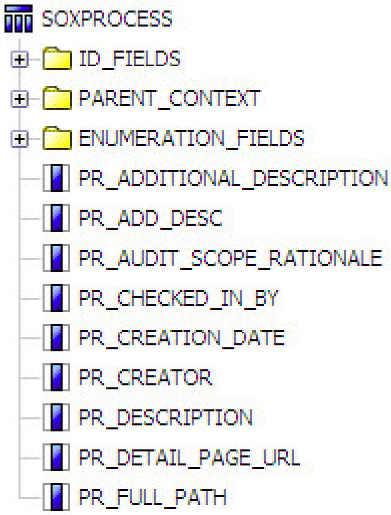
The SOXBUSENTITY_GPC query subject contains three levels of hierarchy for query items: grandparent, parent, and child. The following table describes when to use each level for in a report.

Use this level...	If you want to do this...
grandparent	filter or scope

Use this level...	If you want to do this...
parent	group, aggregate, or prompt
child	filter or scope.

The Object Folder (Relational Model)

Figure 18 shows an example of an object folder with all of its items.



- SOXPROCESS
  - ID_FIELDS
  - PARENT_CONTEXT
  - ENUMERATION_FIELDS
  - PR_ADDITIONAL_DESCRIPTION
  - PR_ADD_DESC
  - PR_AUDIT_SCOPE_RATIONALE
  - PR_CHECKED_IN_BY
  - PR_CREATION_DATE
  - PR_CREATOR
  - PR_DESCRIPTION
  - PR_DETAIL_PAGE_URL
  - PR_FULL_PATH

Figure 18. Object Folder

Table 23 lists the items in an object folder.

Table 23. Object Folder

Folder/Query Subject	Description
ID_FIELDS	These are all the ID fields that are associated with this object.
PARENT_CONTEXT	This folder contains the relationships between the objects and all possible parents. Items in this folder are used to create joins in Report Studio.
ENUMERATION_FIELDS	This folder contains query subjects for the enumerated variables associated with this object.
<Query Items>	These are all of the remaining query items that can be used in a report.

Enumeration Fields Query Items (Relational Model)

The following table lists the query items found in a single value enumerated string:

Table 24. Single Value Enumerated String

Query Subject	Description
<Query Item>_SYSTEM_NAME	This is the system name for the field.

Table 24. Single Value Enumerated String (continued)

Query Subject	Description
<Query Item>	This is the localized name for the field.
<Query Item>_ID	This is the ID of the field.

When displaying an enumerated string value in a report <Query Item> should always be used because it is localized. When filtering on an enumerated string value, use the <Query Item>_ID field because it is indexed.

The following table lists the query items found in a multivalued enumerated string:

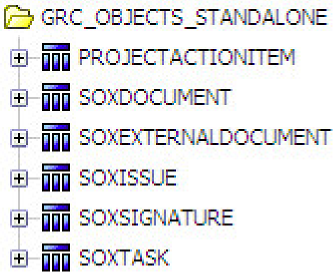
Table 25. Multivalued Enumerated String

Query Subject	Description
<Query Item>_SYSTEM_NAME	This will return one record for each value in the enumerated string. It will be returned as a system name.
<Query Item>	This will return one record for each value in the enumerated string. It will be returned as a localized name.
<Query Item>_ID	This is a pipe "|" separated list of the IDs of the enumeration values.
<Query Item>_SYSTEM_LIST	This is a comma separated list of the system string values of the enumerations.
<Query Item>_LIST	This is a comma separated list of the localized string values of the enumerations.

In order to obtain a comma separated list of the translated values of the string, the report author needs to select the <Query Item>_LIST field and use that in the report. This is the best field to use to display the value of a multivalued enumerated string.

The GRC Objects Standalone Folder (Relational Model)

Figure 19 shows the Standalone Objects folder with all of its items.



GRC_OBJECTS_STANDALONE
- PROJECTACTIONITEM
- SOXDOCUMENT
- SOXEXTERNALDOCUMENT
- SOXISSUE
- SOXSIGNATURE
- SOXTASK

Figure 19. Standalone Objects Folder

Table 26 on page 27 is an explanation of the items in the Standalone Objects folder.

Table 26. Standalone Objects Folder

Folder/Query Subject	Description
<Secondary Object Name>	These are query subjects containing all secondary objects defined for use within your system.

Important: By default SOXTask (Action Item) is left outer joined to SOXIssue object because SOXTask can only be joined to the SOXIssue object in the model.

The Secondary Object Folders (Relational Model)

Figure 20 shows an example of a secondary object folder with all of its items.

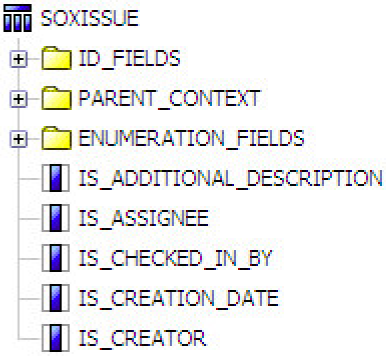


Figure 20. Secondary Object Folder

Table 27 lists the items in a secondary object folder.

Table 27. Secondary Object Folder

Folder/Query Subject	Description
ID_FIELDS	These are all the ID fields that are associated with this object.
PARENT_CONTEXT	This folder contains the relationships between the objects and all possible parents. Items in this folder are used to create joins in Report Studio.
ENUMERATION_FIELDS	This folder contains query subjects for the enumerated variables associated with this object.
<Query Items>	These are all of the remaining query items that can be used in a report.

The Miscellaneous Folder (Relational Model)

Figure 21 on page 28 shows the Miscellaneous folder with all of its sub-folders.

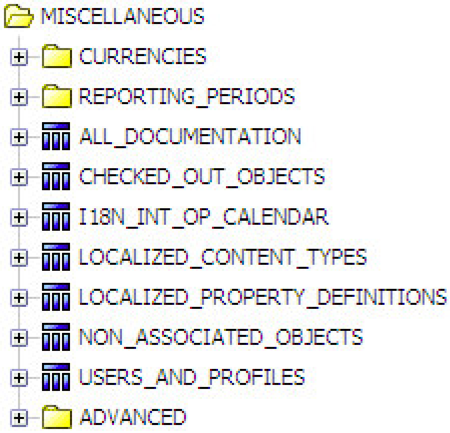


Figure 21. Miscellaneous Folder

Table 28 lists the items in the miscellaneous folder.

Table 28. Miscellaneous Folder

Folder/Query Subject	Description
CURRENCIES	This folder is used to hold query subjects related to system currencies.
REPORTING_PERIODS	This folder is used to hold query subjects related to system reporting periods.
ALL_DOCUMENTATION	This query subject returns a list of all objects in the system and the count/type of documentation that is attached to each object. The following prompts are defined for this query subject: • Entity ID • Set ID (always set = 1)

CHECKED_OUT_OBJECTS	This query object is used to find all objects that are checked out.
	The following prompts are defined for this query subject:
	• Sort Order
	Sort order is an optional prompt that will accept one of the following values:
	`[NAME] = 3`
	`[FULL_PATH] = 4`
	`[CONTENT_TYPE] = 5`
	`[DESCRIPTION]= 8`
	`[CHECKED_OUT_BY] = 9`
	`[CHECKED_OUT_DATE] = 10`
	`[PARENT_NAME] = 11`
	`[PARENT_FULL_PATH] = 12`
	`[PARENT_CONTENT_TYPE] = 13`
	Note: If no sort value is selected, the value defaults to 3 (the `NAME` query item).

Table 28. Miscellaneous Folder (continued)

Folder/Query Subject	Description
I18N_INT_OP_CALENDAR	This table holds the definition of all dates in the system translated to other date values. By creating a relationship with a date field, you can filter the field by the date parts in this query subject.
LOCALIZED_CONTENT_TYPES	This query subject returns the list of all object types defined in the system.
LOCALIZED_PROPERTY_DEFINITIONS	This query subject returns the definition of all properties defined in the system.

NON_ASSOCIATED_OBJECTS	This query subject returns all orphaned objects in your system.
	The following prompts are defined for this query subject:
	• Sort Order
	• Set ID (always set = 1)
	Sort order is an optional prompt that will accept one of the following values:
	`[NAME]= 2`
	`[FULL_PATH] = 3`
	`[CONTENT_TYPE] = 4`
	`[DESCRIPTION] = 7`
	`[CREATOR] = 8`
	`[CREATION_DATE] = 9`
	Tip: If no sort value is selected, the value defaults to 2 (the `NAME` query item).

USERS_AND_PROFILES	This query subject contains the relationship of users to their login profile.
ADVANCED	This folder contains advanced query subjects that have been created to create complex reports.

The Currencies Folder (Relational Model)

Figure 22 shows the Currencies folder with all of its query subjects.

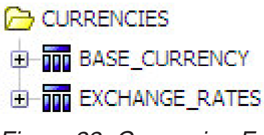


Figure 22. Currencies Folder

Table 29 lists the items in the Currencies folder.

Table 29. Currencies Folder

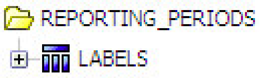
Query Subject	Description
BASE_CURRENCY	This query subject returns the definition of the base currency code in your system.

Table 29. Currencies Folder (continued)

Query Subject	Description
EXCHANGE_RATES	This query subject returns a list of all the defined currencies and their exchange rates.

The Reporting Period Folder (Relational Model)

Figure 23 shows the Reporting Periods folder with all of its query subjects.



```
REPORTING_PERIODS
  LABELS
```

Figure 23. Reporting Period Folder

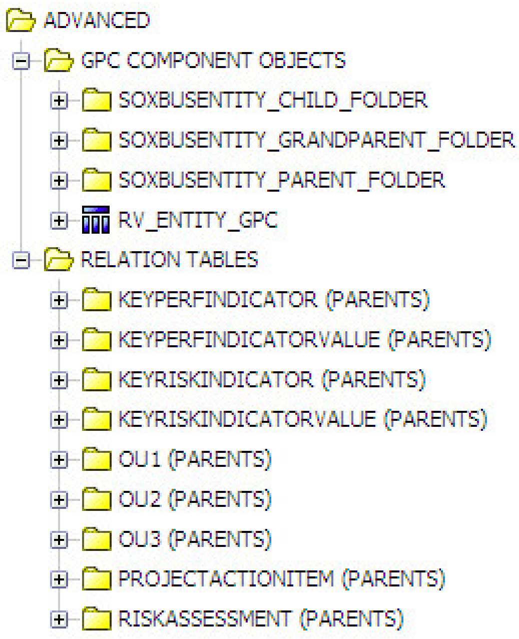
Table 30 lists the items in the Reporting Periods folder.

Table 30. Reporting Period Folder Query Subjects

Query Subject	Description
LABELS	This query subject returns the list of all reporting periods defined in your system.

The Advanced Folder (Relational Model)

Figure 24 shows the Advanced folder with all of its sub-folders.



```
ADVANCED
  GPC COMPONENT OBJECTS
    SOXBUSENTITY_CHILD_FOLDER
    SOXBUSENTITY_GRANDPARENT_FOLDER
    SOXBUSENTITY_PARENT_FOLDER
    RV_ENTITY_GPC
  RELATION TABLES
    KEYPERFINDICATOR (PARENTS)
    KEYPERFINDICATORVALUE (PARENTS)
    KEYRISKINDICATOR (PARENTS)
    KEYRISKINDICATORVALUE (PARENTS)
    OU1 (PARENTS)
    OU2 (PARENTS)
    OU3 (PARENTS)
    PROJECTACTIONITEM (PARENTS)
    RISKASSESSMENT (PARENTS)
```

Figure 24. Advanced Folder

Table 31 lists the items in the Advanced folder.

Table 31. Advanced Folder

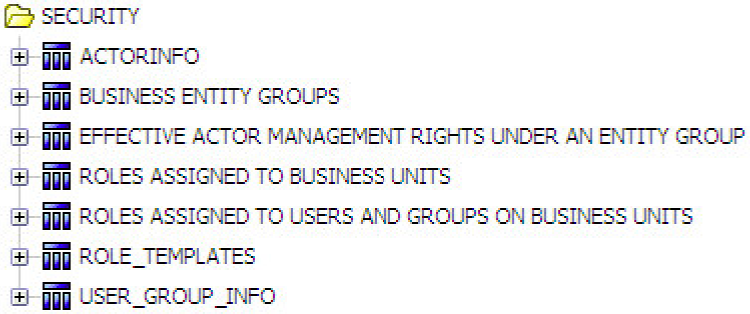
Folder/Query Subject	Description
GPC COMPONENT OBJECTS	This folder contains the GPC query subject split into levels.

Table 31. Advanced Folder (continued)

Folder/Query Subject	Description
RV_ENTITY_GPC	This query subject exposes the view of the GPC object to the report author. It has IBM OpenPages GRC on Cloud data security built into it.
<Object> (PARENTS)	These folders contain bridge tables between objects that can be used for creating advanced joins in a report.

The Security Folder (Relational Model)

Figure 25 shows the Security folder with all of its query subjects.



SECURITY
- ACTORINFO
- BUSINESS ENTITY GROUPS
- EFFECTIVE ACTOR MANAGEMENT RIGHTS UNDER AN ENTITY GROUP
- ROLES ASSIGNED TO BUSINESS UNITS
- ROLES ASSIGNED TO USERS AND GROUPS ON BUSINESS UNITS
- ROLE_TEMPLATES
- USER_GROUP_INFO

Figure 25. Security Folder

Table 32 lists the items in the Security folder.

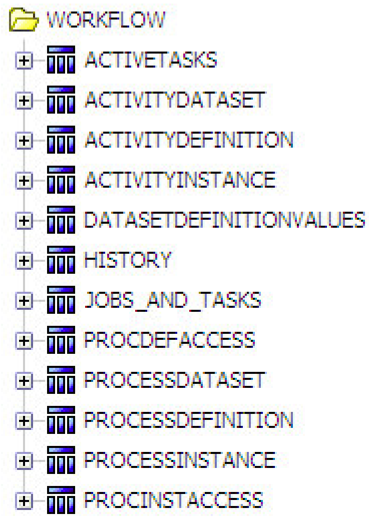
Table 32. Security Folder Query Subjects

Query Subject	Description
ACTORINFO	This query subject returns the definition of all users in the system.
BUSINESS ENTITY GROUPS	This query subject returns the list of entities in the system and their full path.
EFFECTIVE ACTOR MANAGEMENT RIGHTS UNDER AN ENTITY GROUP	This query subject returns a list of entity groups and the effective rights defined for that group. The following prompts are defined for this query subject: • Entity Group ID
ROLES ASSIGNED TO BUSINESS UNITS	This query subject returns a list of all roles assigned to an entity.
ROLES ASSIGNED TO USERS AND GROUPS ON BUSINESS UNITS	This query subject returns a list of all roles assigned to users and groups on an entity.
ROLE_TEMPLATES	This query subject returns a list of all role templates.
USER_GROUP_INFO	This query subject returns a list of all users and groups defined in the system.

The Workflow Folder (Relational Model)

Note: Some of the workflow functionality described in this topic might not be available in IBM OpenPages GRC on Cloud.

Figure 26 shows the Workflow folder with all of its query subjects.



- WORKFLOW
  - ACTIVETASKS
  - ACTIVITYDATASET
  - ACTIVITYDEFINITION
  - ACTIVITYINSTANCE
  - DATASETDEFINITIONVALUES
  - HISTORY
  - JOBS_AND_TASKS
  - PROCDEFACCESS
  - PROCESSDATASET
  - PROCESSDEFINITION
  - PROCESSINSTANCE
  - PROCINSTACCESS

Figure 26. Workflow Folder

Table 33 lists the items in the Workflow folder.

Table 33. Workflow Folder Query Subjects

Query Subject	Description
Activetasks	This table represents each active task (work item) that is a user's representation of an activity instance. Every user that is a valid user for a given activity instance, gets his/her own private copy of the activity instance, which is the active task.
ACTIVITYDATASET	This table represents the data set associated with an activity instance.
ACTIVITYDEFINITION	This table represents an activity definition of a workflow. Activity definitions serve as templates for the creation of activity instances of process instances.
ACTIVITYINSTANCE	An activity instance represents a single activity in a process instance.
DATASETDEFINITIONVALUES	This table represents the definitions of the data elements associated with each process definition.
HISTORY	This table maps an event in a workflow. Since every action performed within a workflow is an event, this table constitutes the exact state of the workflow at all times. Each row in the table represents a workflow event.
JOBS_AND_TASKS	This table represents jobs and their associated task definitions.

Table 33. Workflow Folder Query Subjects (continued)

Query Subject	Description
PROCDEFACCESS	This table represents the internal structure of the process definition.
PROCESSDATASET	This table stores the values of worklist UDAs.
PROCESSDEFINITION	This table represents a workflow process definition.
PROCESSINSTANCE	This table represents a workflow process instance.
PROCINSTACCESS	This table represents the internal structure of the process instance.

The Dimensional Model Top Level Folders

The folder structure in the dimensional model namespace is different from that of the relational model namespace.

Figure 27 shows the dimensional model top level folder layout.

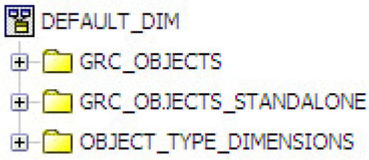


Figure 27. Dimensional Model Folder Layout

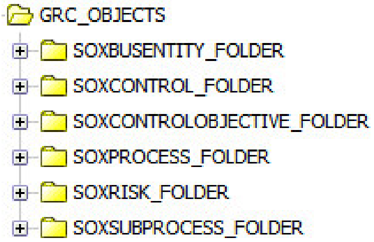
Table 34 lists the items in the dimensional model folder.

Table 34. Top-Level Folder Name and Description

This Top-Level Folder...	Contains Query Subjects for...
GRC_OBJECTS	The dimensions and facts for the primary objects.
GRC_OBJECTS_STANDALONE	The dimensions and facts for the secondary objects.
OBJECT_TYPE_DIMENSIONS	The definitions of all user defined object type dimensions.

The GRC Objects Folder (Dimensional Model)

Figure 28 on page 34 shows the GRC Objects folder with all of its sub-folders.



```
GRC_OBJECTS
  SOXBUSENTITY_FOLDER
  SOXCONTROL_FOLDER
  SOXCONTROLOBJECTIVE_FOLDER
  SOXPROCESS_FOLDER
  SOXRISK_FOLDER
  SOXSUBPROCESS_FOLDER
```

Figure 28. Compliance Objects Folder

Table 35 lists the folders in the GRC folder.

Table 35. GRC Objects Folders

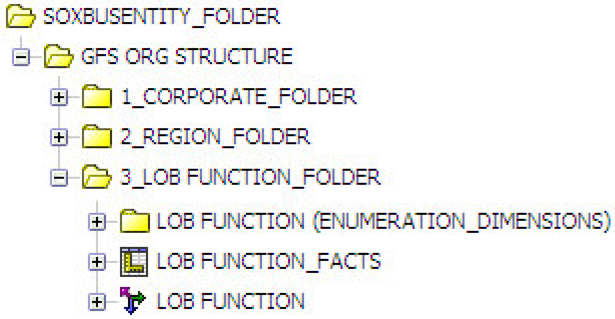
Folder	Description
SOXBUSENTITY_FOLDER	This folder contains all of the dimensions and facts related to the business entity object.
<Object>_FOLDER	These folders contain all of the dimensions and facts related to the objects specified in this namespace.

The SOXBusEntity Folder (Dimensional Model)

The SOXBusEntity folder contains folders for the custom recursive object levels. The folders reflect the names and levels assigned using the Business Entity recursive object feature in IBM OpenPages GRC on Cloud.

This folder will be different for each customer depending on how each entity recursive object level was defined.

Figure 29 shows an example of a SOXBusEntity folder with all of its items.



```
SOXBUSENTITY_FOLDER
  GFS ORG STRUCTURE
    1_CORPORATE_FOLDER
    2_REGION_FOLDER
    3_LOB FUNCTION_FOLDER
      LOB FUNCTION (ENUMERATION_DIMENSIONS)
      LOB FUNCTION_FACTS
      LOB FUNCTION
```

Figure 29. SOXBusEntity Folder

Table 36 lists the folders in the SOXBusEntity folder.

Table 36. Entity Folder

Folder	Description
SOXBUSENTITY_FOLDER	The SOXBusEntity folder used to hold object type dimensions created by the user.
GFS ORG STRUCTURE	The name of the object type dimension created in OpenPages GRC on Cloud.

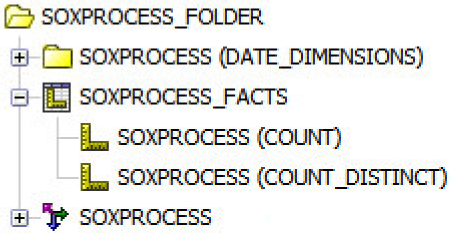
Table 36. Entity Folder (continued)

Folder	Description
<Recursive Object Level>	For each level of the object type dimension, this folder holds the definition of a single level.
<Recursive Object Level> (ENUMERATION_DIMENSIONS)	The enumeration dimensions defined for this level in the recursive object.
<Recursive Object Level>_FACTS	The facts defined for this level in the recursive object.
<Recursive Object Level>	The remaining attributes defined for this level of the recursive object are contained in this dimension.

The Primary Object Folder (Dimensional Model)

The primary objects folder will be different for each customer depending on the objects that have been defined for your system through IBM OpenPages GRC on Cloud. Please see the *IBM OpenPages GRC on Cloud Administrator's Guide* for more information on how to configure facts and dimensions for each object.

Figure 30 shows an example of a primary object folder.



SOXPROCESS_FOLDER
⊞ SOXPROCESS (DATE_DIMENSIONS)
⊟ SOXPROCESS_FACTS
SOXPROCESS (COUNT)
SOXPROCESS (COUNT_DISTINCT)
⊞ SOXPROCESS

Figure 30. Primary Object Folder

Table 37 is an explanation of the folders in a primary objects folder.

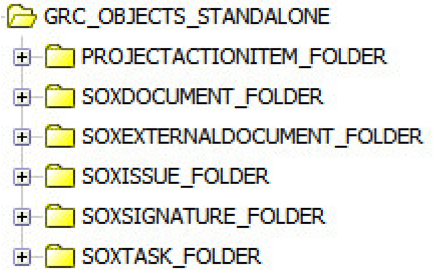
Table 37. Object Folders

Folder	Description
<Object Name> (DATE_DIMENSIONS)	A folder containing all of the date dimensions for each object.
<Object Name>_FACTS	The facts defined for the object.
<Object Name>	A dimension defined for each object that contains all the remaining fields. These fields are typically used when creating a dimension based list report.

The GRC Standalone Objects Folder (Dimensional Model)

The GRC Standalone Object folder will be different for each customer depending on which secondary objects are defined.

Figure 31 shows an example of the GRC Standalone Objects folder with all of its sub-folders.



GRC_OBJECTS_STANDALONE
- PROJECTACTIONITEM_FOLDER
- SOXDOCUMENT_FOLDER
- SOXEXTERNALDOCUMENT_FOLDER
- SOXISSUE_FOLDER
- SOXSIGNATURE_FOLDER
- SOXTASK_FOLDER

Figure 31. GRC Standalone Objects Folder

Table 38 is an explanation of the folders in the GRC Standalone Objects folder.

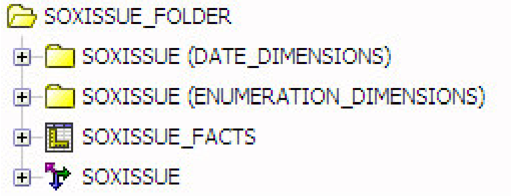
Table 38. Standalone GRC Objects Folders

Folder	Description
<Object Name>_FOLDER	A folder containing all of the measures and dimensions for each object.

The Object Folders (Dimensional Model)

This folder will be different for each customer depending on the objects that have been defined for your system through IBM OpenPages GRC on Cloud. Please see the *IBM OpenPages GRC on Cloud Administrator's Guide* for more information on how to define facts and dimensions for an object.

Figure 32 shows an example of an object folder.



SOXISSUE_FOLDER
- SOXISSUE (DATE_DIMENSIONS)
- SOXISSUE (ENUMERATION_DIMENSIONS)
- SOXISSUE_FACTS
- SOXISSUE

Figure 32. Object Folder

Table 39 is an explanation of the folders an object folder.

Table 39. Object Folders

Folder	Description
<Object Name> (DATE_DIMENSIONS)	A folder containing all of the date dimensions related to the object.
<Object Name> (ENUMERATION_DIMENSIONS)	A folder containing all of the enumeration dimensions related to the object.
<Object Name>_Facts	The facts defined for this object.

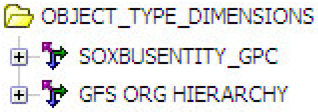
Table 39. Object Folders (continued)

Folder	Description
<Object Name>	A dimension defined for each object that contains all the remaining fields. These fields are typically used when creating a dimension based list report.

The Object Type Dimensions Folder (Dimensional Model)

The Object Type Dimension folder will be different for each customer depending on the object type dimensions that have been defined for your system through IBM OpenPages GRC on Cloud. Please see the *IBM OpenPages GRC on Cloud Administrator's Guide* for more information on how to create these dimensions.

Figure 33 shows an example of the Object Type Dimensions folder.



OBJECT_TYPE_DIMENSIONS
SOXBUSENTITY_GPC
GFS ORG HIERARCHY

Figure 33. Object Type Dimensions Folder

Table 40 is an explanation of the folders in the Object Type Dimension folder.

Table 40. Object Type Dimensions Folders

Folder	Description
SOXBUSENTITY_GPC	This is a default object type dimension that creates a GPC dimension. This object is used for filtering only! It cannot be used for drill-up/drill-down functionality.
GFS ORG HIERARCHY	This is a custom object type dimension that was entered through OpenPages GRC on Cloud to create a hierarchy for business entities.

Chapter 5. Optimizing Performance

This chapter lists techniques that report authors can use to improve report performance.

Filtering Top-level Business Entities

To improve the performance of a report, all reports should either have a prompt that filters to a specific business entity or a filter that scopes the report by the top-level business entity or entities. If reports are run without being scoped to a single entity, the query performs multiple searches through the hierarchy. This slows down the response time greatly and may introduce undesirable results for the entity prompt query page or report.

The prompt can take any of the following forms:

- A single prompt displaying the full path of the valid entities the user can select from.
- A series of cascaded prompts that enables the user to work their way down to the exact entity they want to report from.
- A tree prompt that enables filtering on a dimensional report.

If a prompt is not required because the report will always be run from the same entity, then a fixed filter can be placed in the query subject that always selects the exact entity.

Note: Do not use Resource IDs as fixed filters as they will be different on every installation.

Filtering on Reporting Periods

All reports should include a filter on the reporting period. A reporting period is a snapshot of all of the data in your database as a function of time. This creates a very large data set each time this operation is performed.

The filter can be created using a prompt value from a prompt page or by creating a reporting period filter that only selects (for instance) the current reporting period:

```
[DEFAULT_REL].[SOXBUSENTITY_GPC].[GEN_REPORTING_PERIOD_ID] = -1
```

The value of -1 signifies the current reporting period.

Bypassing an Index

In the database tables, indexes are defined on all the enumerated string value IDs. If you use enumerated string value IDs as filters, you may want to bypass the specific index on the field for better performance overall.

When you create an enumerated string value filter, what occurs is that instead of choosing a system index to join two object tables (that include the string filter), Oracle will choose less efficient indexes due to the enumerated string filter. By bypassing the index on the enumerated string, Oracle will create a faster query.

For example, modify the filter on the enumerated string value "Effective" as follows:

```
[DEFAULT_REL].[SOXCONTROL].[CN_OPERATING_EFFECTIVENESS_ID] + 0 =
#$SOXCONTROL_OPERATING_EFFECTIVENES_DEFINITION_MAP{"Effective"}#
```

When you add a 0 (zero) to the left side of the filter expression, Oracle treats it as an expression and the index attached to that field will be ignored.

It is often difficult to see the performance improvement as a result of this change. It is more apparent when a large number of rows are returned either when you output to a PDF/Excel file or the row return value is set to a large number for the data container instead of the standard value of 20 rows.

Note: This performance enhancement technique is applicable only for filters that contain enumerated string IDs.

Query Direction Performance

When exploring all the computation possibilities, there is one large distinction in what can/should be done. When writing a report, it is possible to query up the relationship tree (that is, compute values based on ancestors) and this activity is strongly discouraged. The automatic framework generation is set up in such a way as to create joins that are conducive to better performance querying down the relationship tree. A query up the tree will result in bad computed field performance as well as place a large strain on the database that can result in the entire application slowing down.

Adding New Indexes

If you find that a pattern in your reports involves joining two fields that are not indexed, it is worthwhile investigating whether adding the index will improve the report's performance. If it does, you can add the index permanently through IBM OpenPages GRC on Cloud. Please see the *IBM OpenPages GRC on Cloud Administrator's Guide* for adding indexes to tables.

The Use for Parameter Info Setting

There is a property on all query subjects called **Use for Parameter Info**. What this setting does is determine which queries get priority when determining parameter information. So for prompt queries, this value should be set to **Yes**. For all other queries, this parameter should be set to **No**. This setting only improves prompt page performance.

Recursive Object Levels Root Binding

When using the dimensional model you typically must create Recursive Object Levels to support a business entity hierarchy.

The Starting Entity is set to / which is the root of the business entity hierarchy in your IBM OpenPages GRC on Cloud. By making this selection, this dimension will pull in all business entities in your system into your report. This might be what you are looking for but typically at the top level of your business entity definitions are your legal entities, audit library, general library, business entities, etc. which are not necessarily what you want to see in a single report.

In order to make the dimension you define have better performance, select a Starting Entity that matches what you want to display. For example, select Global Financial Services.

You will notice the following changes:

- The Starting Entity is now /Global Financial Services. Since the reports are typically on your business entity reporting hierarchy, this is the best place to start from.
- This will improve performance by leaving out all the other top level business entities that were not really required for reporting in this particular dimension.

Chapter 6. Report Design Language

Report Design Language (RDL) is the behind the scenes language that all of your reports are stored in. The result is that a report can be completely defined in XML and its source content can be viewed in any XML editor.

Beginning with IBM OpenPages 6.0 you are no longer required to design your reports in RDL. However, it is still recommended that report authors use RDL to design their reports due to the limitations listed below when not using RDL.

The following are the list of limitations if you do not develop your report in RDL:

- If you change a field label in IBM OpenPages GRC on Cloud, it will be displayed properly as a new insertable object but if you have already used it in your report, that reference will not be updated to the new name. This will lead to confusion.
- Dimensional drill-up/drill-down no longer functions.
- CQS files will still be in RDL.
- The expression editor will still only display in RDL.
- Fields in the framework model are sorted by their name in RDL.
- It will be more difficult to support a report that has a mixture of native language and RDL fields.

Setting Your Locale in OpenPages GRC on Cloud

All users are required to have a default locale in IBM OpenPages GRC on Cloud. This determines what language the application is displayed in along with what language you see when a report is executed.

For more information on configuring your locale, see the *IBM OpenPages GRC on Cloud Administrator's Guide*.

Language and locale support

If you are using IBM OpenPages GRC on Cloud in a language other than English, this information will help you to understand the language and locale settings.

Web browser language preference

The web browser language preference is the setting that you choose to specify the language that web pages can be displayed in. The web browser language preference affects only the OpenPages GRC on Cloud login page. The web browser language preference does not affect number and date formatting in OpenPages GRC on Cloud.

If the web browser language preference is set to a language other than one of the following languages, be aware that the OpenPages GRC on Cloud login page appears in English:

- German
- Spanish
- French
- Italian

- Japanese
- Portuguese
- Chinese
- English

Locale setting

The **Locale** list contains a list of product languages. This language setting controls the language of the product except for the login page.

Data formatting and report languages are available in the following cultures in the **Locale** list:

Table 41. Languages in the **Locale** *list and the cultures that they represent*

Language in the Locale list	Culture
French	French (France)
German	German (Germany)
Italian	Italian (Italy)
Japanese	Japanese (Japan)
Portuguese	Portuguese (Brazil)
Spanish	Spanish (Spain)
Simplified Chinese	Chinese (China)
Traditional Chinese	Chinese (Taiwan)
U.K. English	English (UK)
U.S. English	English (US)

Considerations for specific languages

When OpenPages GRC on Cloud is set to use U.S. English, dates are formatted as mm/dd/yy. For example, January 3, 2013, is formatted as 1/3/13 rather than 03/01/2013 in U.K. English.

When the product is set to use Spanish (Spain), numbers are formatted as 123.456,78, where the period is a thousands separator and the comma is used as a decimal separator. For example, the number twelve thousand and five hundred is formatted as 12.500 in Spanish (Spain) rather than 12,500 in Spanish (Mexico).

In several cultures, the convention is to place the currency symbol to the right of the number. In OpenPages GRC on Cloud, currency symbols are always displayed to the left of the number.

Date formatting can be unconventional as well.

Related concepts:

"Localization Display Issues" on page 259

Setting Your Locale in Cognos Connection

By setting your locale in Cognos Connection, you still maintain your ability to run the IBM OpenPages GRC on Cloud application in the locale specified within the application along with the added capability to design reports in RDL.

Procedure

1. Login to Cognos Connection.
2. Select **My Home** from the main screen.
3. Locate the Cognos Connection toolbar.
4. Select **My Preferences** from the **My Area** drop down list in the portal.
5. Select the **General Tab**.
6. Set the **Content Language** to **English (Canada)**. English (Canada) is actually Report Design Language.
7. Click **OK**.

Chapter 7. Working with Relational Data

This chapter explains how to use the reporting framework when working with relational data.

- For a list of known issues and any available workarounds, see Appendix C, “Limitations and Workarounds,” on page 259
- For a list of performance tips and tricks, see Chapter 5, “Optimizing Performance,” on page 39

Syntax Conventions

Field Expression Syntax

The examples in this chapter use the following syntax for filter or field expressions:

```
[NAMESPACE].[QUERY SUBJECT].[QUERY ITEM] = value
```

For example,

```
[DEFAULT_REL].[SOXCONTROL].[CN_OPERATING_EFFECTIVENES] = 'Effective'
```

`[DEFAULT_REL]` is a namespace in the predefined default object hierarchy. If you are using a different namespace, then substitute the name of that namespace for the default namespace.

Folder Navigation Syntax

The examples in this chapter use the following syntax to show navigation to a query item under the `DEFAULT` namespace:

FOLDER|[*QUERY SUBJECT*]|[*QUERY ITEM*]

For example,

```
REPORTING_PERIODS|[LABELS]|[NAME]
```

Working With Data Containers

The following sections describe how to create and format list, crosstab, and graph reports. For details on modifying lists, crosstabs, and charts, see the *IBM Cognos 10 Business Intelligence Report Studio User Guide*.

Working with Lists

A list object is used to create a report that will show data in rows and columns. When you first create a list object, it appears empty.

The header row is highlighted in blue while the column data is represented by the white. When a data item is placed into a list both the column header and column data are filled in.

The list itself is divided into the following areas:

- Individual Cells - Controls the format
- List Column Body - Controls the formatting of the data area of an individual column.

- List Column - Controls the formatting of an individual column.
- List Columns Body Style - Controls the formatting of the data area of all the columns.
- List Columns Title Style - Controls the formatting of the headers of all the columns.
- List Columns - Controls the formatting of the list columns.
- List - Controls the formatting of the entire list.

Each of the above items has a set of properties that can be set to affect how the data appears to the user. Each item has a slightly different set of properties.

Selecting Data Columns

Follow this procedure to select data columns for the list report.

Procedure

1. From the Insertable Objects pane in the **Source** tab, drag and drop the following data items into the list:
   - `DEFAULT|[DEFAULT_REL]|GRC_OBJECTS|[SOXCONTROL]|[CN_NAME00]`
   - `DEFAULT|[DEFAULT_REL]|GRC_OBJECTS|[SOXCONTROL]|[CN_DESCRIPTION]`
2. Run the report.

   When you drop items into a list, a query is formed using the data items you selected.

Results

If you don’t like the order of the columns, you can select a column and move it with the mouse to elsewhere in the list.

Working with Crosstabs

Crosstabs are a type of data container that operates on aggregated data. The row and column fields in the crosstab are typically string fields or enumerations while the intersection of the crosstab must always be a summarized piece of data, typically a numeric value.

You will see three areas in the crosstab object:

- Columns - Data items placed here will generate column headers. Multiple data items can be dropped here to create nested column headers.
- Rows - Data items placed here will generate the rows of data in the crosstab. Multiple data items can be placed here to generate nested rows.
- Measures - Only a single data item can be placed here that will be aggregated against the rows and columns selected.

Selecting Data Items

Follow this procedure to select which data items you want to include as rows and columns in the crosstab report, and how you want them formatted.

Procedure

1. Using the **Query Explorer**, select **Query1**.
2. From the Insertable Objects pane, drag and drop the following data items into the data items pane of the crosstab:
   - `DEFAULT|[DEFAULT_REL]|GRC_OBJECTS|[SOXCONTROL]|ID_FIELDS|[CN_CONTROL_ID]`

- `DEFAULT|[DEFAULT_REL]|GRC_OBJECTS|[SOXCONTROL]|ENUMERATION_FIELDS|OPERATING_EFFECTIVENES (ENUMERATION)|[CN_OPERATING_EFFECTIVENES]`
- `DEFAULT|[DEFAULT_REL]|GRC_OBJECTS|SOXBUSENTITY_FOLDER|[SOXBUSENTITY_GPC]|[SOXBUSENTITY_GPC]|[CEN_NAME00]`

3. Drag and drop the [CN_OPERATING_EFFECTIVENES] into the columns area from the **Data Items** tab of the Insertable Objects pane.
4. Drag and drop the [CEN_NAME00] into the rows area from the **Data Items** tab of the Insertable Objects pane.
5. Drag and drop the [CN_CONTROL_ID] into the measures area from the **Data Items** tab of the Insertable Objects pane.
6. Select the measures area and in the Properties pane, do the following:
   a. Under the **Data Item**, set the **Aggregate Function** to **Count Distinct**.
   b. Set the **Rollup Aggregate Function** to **Automatic**.
7. Run the report.

Results

Click on the rows column and using the **Sort** icon, to sort the rows in ascending order.

Click on the **Summation** icon and select **Total** to add summaries to the rows and columns.

Working with Graphs

Graphs are used to display data in a visual format. There are a variety of graph formats built into CommandCenter that the report author should become familiar with in order to best display data to the end users.

A graph consists of the following components:

- Default Measure - This is typically the y-axis component of the graph. It will be a numerical value.
- Series - A group or groupings for the y-axis.
- Categories - This is typically the x-axis component of the graph.

Creating a Graph

Follow this procedure to create a graph.

Procedure

1. Create a new Chart report.
   a. Select the **Pie, Donut** Chart grouping.
   b. Select the **Pie** chart type.
   c. Click **OK**.
2. Drag and drop the following query items into the various chart sections:
   - `Categories (pies)`

     `DEFAULT|[DEFAULT_REL]|GRC_OBJECTS|SOXBUSENTITY_FOLDER|[SOXBUSENTITY_GPC]|[SOXBUSENTITY_GPC]|[CEN_NAME00]`
   - `Default Measures`

     `DEFAULT|[DEFAULT_REL]|GRC_OBJECTS|[SOXCONTROL]|ID_FIELDS|[CN_CONTROL_ID]`
   - `Series (pie slices)`

```
DEFAULT|[DEFAULT_REL]|GRC_OBJECTS|[SOXCONTROL]|ENUMERATION_FIELDS|
OPERATING_EFFECTIVENES (ENUMERATION)|[CN_OPERATING_EFFECTIVENES]
```

3. From the **Query Explorer**, select the query.
   a. In the Properties pane under **Miscellaneous**, set the name of the query to **chartMain**.
4. In the **Data Items** pane, select `[CN_CONTROL_ID]`**.**
   a. In the Properties pane under **Data Item**, change the value of the **Aggregate Function** property to **Count Distinct**.
   b. In the Properties pane under **Data Item**, change the value of the **Rollup Aggregate Function** property to **Automatic**.
5. From the **Page Explorer**, create a **Prompt Page** and create a **Business Entity** prompt as shown in "Adding a Business Entity Prompt" on page 56.
6. Return to the main report page.
7. Double-click on the title and set the value to **Operating Effectiveness**.
8. Run the report.

Working With Enumerated Fields

The following sections describe how to include enumerated fields in reports.

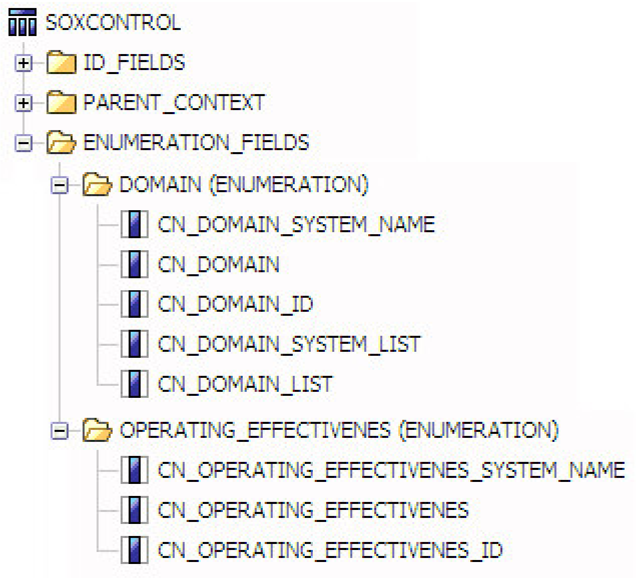
- "The Enumerated Fields Folder Structure"
- "Using Enumerated Fields in a Report" on page 51

The Enumerated Fields Folder Structure

Each object has a folder containing all the enumerated fields defined for that object. Figure 34 on page 51 shows the SOXControl query subject with its enumerated fields sub-folder. This folder contains the object specific values of the enumerated fields.

As was previously mentioned there is a second enumerated fields folder in the framework. This folder holds the definition of the enumeration to be used for prompts. See Chapter 4, "Reporting Framework," on page 15 for more information on the other folder.

In Figure 34 on page 51 we are choosing the folder that holds the object specific enumeration values.



- SOXCONTROL
  - ID_FIELDS
  - PARENT_CONTEXT
  - ENUMERATION_FIELDS
    - DOMAIN (ENUMERATION)
      - CN_DOMAIN_SYSTEM_NAME
      - CN_DOMAIN
      - CN_DOMAIN_ID
      - CN_DOMAIN_SYSTEM_LIST
      - CN_DOMAIN_LIST
    - OPERATING_EFFECTIVENES (ENUMERATION)
      - CN_OPERATING_EFFECTIVENES_SYSTEM_NAME
      - CN_OPERATING_EFFECTIVENES
      - CN_OPERATING_EFFECTIVENES_ID

Figure 34. SOXControl Enumerated Fields

There are two types of enumerated fields: single valued and multivalued. The single valued field has the following query items:

- <Query Item>_SYSTEM_NAME - This is the system name for the field.
- <Query Item> - This is the localized name for the field.
- <Query Item>_ID - This is the ID of the field.

When displaying the value in a report <Query Item> should always be used. When filtering on this value, use the <Query Item>_ID field because it is indexed.

Multivalued fields have the following query items:

- <Query Item>_SYSTEM_NAME - This will return one record for each value in the enumerated string. It will be returned as a system name.
- <Query Item> - This will return one record for each value in the enumerated string. It will be returned as a localized name.
- <Query Item>_ID - This is a pipe "|" separated list of the IDs of the enumeration values.
- <Query Item>_SYSTEM_LIST - This is comma separated list of the system string values of the enumerations.
- <Query Item>_LIST - This is comma separated list of the localized string values of the enumerations.

In order to obtain a comma separated list of the translated values of the string, the report author needs to select the <Query Item>_LIST field and use that in the report. This is the best field to use to display the value of a multivalued enumerated string.

Using Enumerated Fields in a Report

Follow this procedure to use an enumerated field in a report.

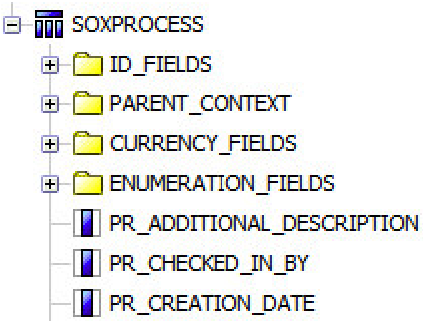
Procedure

1. Click the **Source** tab on the Insertable Objects pane and select the query item you want to use in the report.
2. Drag and drop the query item into the report.

Working with Date Fields

Date fields are listed as query items attached to the object.

In Figure 35 you can see the [PR_CREATION_DATE] listed as a query item.



SOXPROCESS
- ID_FIELDS
- PARENT_CONTEXT
- CURRENCY_FIELDS
- ENUMERATION_FIELDS
- PR_ADDITIONAL_DESCRIPTION
- PR_CHECKED_IN_BY
- PR_CREATION_DATE

Figure 35. SOXProcess Query Subject

Follow this procedure to use a date field in a report.

Procedure

1. Click the **Source** tab on the Insertable Objects pane and select the query item you want to use in the report.
2. Drag and drop the query item into the report.
3. Select the query item you just dropped into the report.
4. In the Properties pane under **Data** select **Data Format**.
5. Do the following to format the date field:
   a. Select **Date** for the **Format type**.
   b. Select **Short** for the **Date Style**.
   c. Select **/** for the **Date Separator**.
   d. Select **Month, Day, Year** for the **Date Ordering**.
   e. Select **Show Century** (to show a 4 digit year) for **Display Years**.
   f. Select **Two Digits** for **Display Months**.
   g. Select **Two Digits** for **Display Days**.
6. Click **OK**.

Results

In addition to the manual formatting of dates, there are built-in formats that are localized in accordance with the user's locale. If you do not create a format for a date field, these formats will be in effect. To see the predefined date fields see "Translating Date Fields using Predefined Date Formats" on page 152.

Working With Currency Fields

The following sections describe how to add currency fields to reports.
- "System Currency Query Subjects" on page 53
- "Currency Fields Folder Structure" on page 53
- "Configuring the use of currency fields in a report" on page 54

System Currency Query Subjects

System currency query objects are located under the `MISCELLANEOUS|CURRENCIES` folder in a namespace, and are categorized into either exchange rates or base currency, as shown in Figure 36.

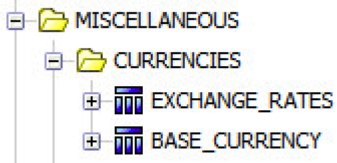

MISCELLANEOUS
CURRENCIES
EXCHANGE_RATES
BASE_CURRENCY

Figure 36. System Currency Folder

The exchange rates query subject includes a record for each exchange rate that was set in the system. The base currency query subject only returns a single record with the base currency information.

Some examples of how you might use currency objects would be to:

- Display the base currency (ISO code) in a report.
- Get the latest exchange rate for a given currency code (as an alternative, you could use the `GET_EX_RATE_ISO` database function). For details, see "Getting the Exchange Rate" on page 181.
- Display a currency symbol instead of a currency code in a report.

Currency Fields Folder Structure

In addition to the above query subjects, each object has a folder containing all the currency fields defined for that object. Figure 37 show the SOXRisk query subject with its currency fields sub-folder.

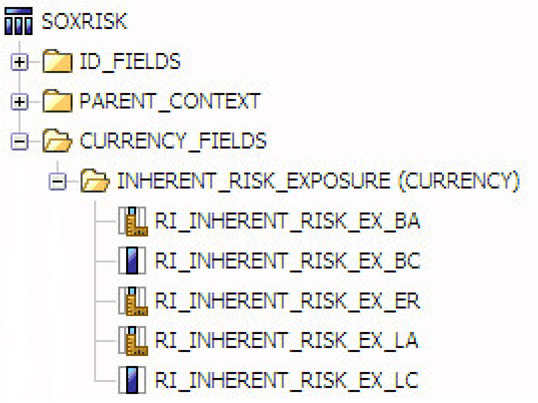

SOXRISK
ID_FIELDS
PARENT_CONTEXT
CURRENCY_FIELDS
INHERENT_RISK_EXPOSURE (CURRENCY)
RI_INHERENT_RISK_EX_BA
RI_INHERENT_RISK_EX_BC
RI_INHERENT_RISK_EX_ER
RI_INHERENT_RISK_EX_LA
RI_INHERENT_RISK_EX_LC

Figure 37. SOX Account Currency Fields

Each currency field will be located in its own folder with five values:

- <Query Item>_BA - The base currency value of the field.
- <Query Item>_BC - The base currency code.
- <Query Item>_ER - The exchange rate.
- <Query Item>_LA - The localized currency value of the field.
- <Query Item>_LC - The local currency code.

Depending on the requirement, you select the appropriate field for your report.

Configuring the use of currency fields in a report

You can use a currency field in a report. Configure the application to allow the use of currency fields.

Procedure

1. Click the **Source** tab on the Insertable Objects pane Select the query item to use in the report.
2. Drag and drop the query item into the report.
3. Select the query item that you just dropped into the report.
4. In the Properties pane under **Data** select **Data Format**.
5. Complete the following actions to format the currency field:
   a. Select **Currency** as the **Format type**.
   b. Either select a predefined currency format (such as $ (USD)) or configure the format of the query item.
6. Click **OK**.

Results

In addition to the manual formatting of currencies, built in formats are included that are localized based on the user locale. If you do not create a format for a currency field, default formats are used. For information on the predefined currency formats, see “Translating Currency Fields using Custom Formats” on page 155.

Adding Prompts

The prompt page adds interactivity for users and acts as a report-scoping mechanism for filtering and limiting the retrieved data.

Creating a Prompt Page

Follow this procedure to create a prompt page in your report.

Procedure

1. Create a prompt page using the **Build Prompt Page** icon on the toolbar.
2. Do the following to the title:
   a. Double click on **Double click to edit text** and enter the title of the report.
   b. In the **Properties Pane** under the **Miscellaneous** category, set the **Classes** property to **empty**.
   c. Remove the title underline by clicking on the **Underline** icon.
3. In the footer, do the following:
   a. Remove any buttons that you don’t think you will need by selecting the button and clicking the **Delete** icon.
   b. Now click the footer and then the **Center** icon to center the buttons on the page.
   c. Now click the remaining buttons and set the **Background Color** property to **Silver** in the Properties pane.
4. Click the **Toolbox** tab on the Insertable Objects pane.

5. Drag a Table **Table** object into the body of the prompt page. Select 2 columns by however many rows you will need. Include a blank row after each prompt.
6. When you have a blank row between controls, you will need to set the default cell height to 15 pt. Do the following:
   a. Select the cell in the table that you want to set the height.
   b. In the Properties pane, under **Positioning**, select **Size and Overflow**.
   c. In the dialog box, set the **Height** to 15 pt.
   d. Click **OK**.

Adding a Reporting Period Prompt

Follow this procedure to add a reporting period prompt to your report.

Procedure

1. If you do not already have a prompt page in your report, follow the instructions in “Creating a Prompt Page” on page 54.
2. To create your Reporting Period prompt, do the following:
   a. Drag a Text Item **Text Item** into the first column of the table. Enter the text for the prompt in the text item, **Reporting Period:**.
   b. Select the table cell with the text item and set the location to be **Horizontally** right and **Vertically** top.
   c. Make the text bold using the B **Bold** icon.
   d. Drag a Value Prompt **Value Prompt** object into the second column on your prompt page. This will bring up the Prompt Wizard.
3. On the **Choose Parameter** screen of the Prompt Wizard:
   a. Select the option **Create a new parameter**.
   b. In the box, type a name for the parameter. For example, **Reporting Period**.
   c. Click **Next**.
4. On the **Create Filter** screen:
   a. Select the option **Create a parameterized filter**.
   b. In the **Model item** box, click the browse button and navigate to the LABELID query item as follows:
      ```
      [DEFAULT_REL]|MISCELLANEOUS|REPORTING PERIODS|[LABELS]|[LABELID]
      ```
   c. The following values will be displayed in the box:
      ```
      [DEFAULT_REL].[LABELS].[LABELID]
      ```
   d. In the **Operator** box, use = (the equal sign is the default).
   e. Clear the box for the **Make this filter optional** option.
   f. If you have multiple queries in your report, in the **Apply filter** box, select the query to which the filter should be applied.
   g. Click **Next**.
5. On the **Populate Control** screen:
   a. Select the **Create new query** option.
   b. In the **Name** box, type a name for the query. For example, **Reporting Period Prompt**.
   c. In the **Values to display** box, click the browse button and navigate to the NAME query item as follows:
      ```
      [DEFAULT_REL]|MISCELLANEOUS|REPORTING_PERIODS|[LABELS]|[NAME]
      ```

d. The following values will be displayed in the box:

   `[DEFAULT_REL].[LABELS].[NAME]`

e. Click **Finish**.

Passing the OpenPages GRC on Cloud Reporting Period Parameter

The "Reporting Period" selector in the user's IBM OpenPages GRC on Cloud can be used to set the reporting period in a report. This requires that when the report is installed in the application, additional configuration is performed to pass the reporting period values to the report.

For example, if a user in the OpenPages GRC on Cloud application selects the current period prior to clicking a report, this reporting period setting is transferred to the report at the time the report is launched.

When a report is launched from the application, the following reporting period parameters are sent:

- `ReportingPeriodId` - contains the value of the identifier for the selected Reporting Period
- `ReportingPeriodName` - contains the value of the name for the selected Reporting Period

To filter reports, you should use the parameter `ReportingPeriodId` in filter expressions. If these parameters are used in the report, then there is no need to create a prompt page for selecting a Reporting Period.

You can take advantage of the user's reporting period setting by creating a filter with the exact syntax below entered into the Expression Definition:

```
[DEFAULT_REL].[SOXBUSENTITY_GPC].[GEN_REPORTING_PERIOD_ID] = ?ReportingPeriodId?
```

Then, the reporting period of the application is used to filter the report. Remember that this filter will have to be added to each query in the report.

Adding a Business Entity Prompt

Follow this procedure to add an Entity prompt to your report.

Procedure

1. If you do not already have a prompt page in your report, follow the instructions in "Creating a Prompt Page" on page 54.
2. To create your Business Entity prompt, do the following:
   a. Drag a **Text Item** into the first column of the table. Enter the text for the prompt in the text item.
   b. Select the table cell and set the location to be horizontally right and vertically top.
   c. Make the text bold using the bold icon.
   d. Drag a **Value Prompt** object into the second column on your prompt page. This will bring up the Prompt Wizard.
3. On the **Choose Parameter** screen of the Prompt Wizard:
   a. Select the option **Create a new parameter**.
   b. In the box, type a name for the parameter. For example, **Business Entity:**.
   c. Click **Next**.
4. On the **Create Filter** screen:

  a. Select the option **Create a parameterized filter**.
  b. In the **Model item** box, click the browse button and navigate to the following query item:

```
[DEFAULT_REL]|GRC_OBJECTS|SOXBUSENTITY_FOLDER|
[SOXBUSENTITY_GPC]|ID_FIELDS|[PEN_ENTITY_ID]
```

  c. In the **Operator** box, use = (the equal sign is the default).
  d. Clear the box for the **Make this filter optional** option.
  e. If you have multiple queries in your report, in the **Apply filter** box, select the query to which the filter should be applied.
  f. Click **Next**.

5. On the **Populate Control** screen:
  a. Select the **Create new query** option.
  b. In the **Name** box, type a name for the query. For example, **Business Entity Prompt** Query.
  c. In the **Values to display** box, click the browse button and navigate to the following query item:

```
[DEFAULT_REL]|GRC_OBJECTS|SOXBUSENTITY_FOLDER|
[SOXBUSENTITY_GPC]|[PEN_FULL_PATH]
```

  d. The following values will be displayed in the box:

```
[DEFAULT_REL].[SOXBUSENTITY_GPC].[PEN_FULL_PATH]
```

  e. Click **Finish**.

6. Now select the prompt you just created and under the **General** category in the Properties pane, change the **Select UI** property to **List Box**.

Adding a Single Select Enumerated Value Prompt

If the report has to be filtered interactively on an enumerated string value, you can use a Value Prompt to perform this action. Follow this procedure to add an enumerated value prompt to your report.

Procedure

1. If you do not already have a prompt page in your report, follow the instructions in "Creating a Prompt Page" on page 54.
2. To create your Enumerated Value prompt, do the following:
  a. Drag a Text Item **Text Item** into the first column of the table. Enter the text for the prompt in the text item.
  b. Select the table cell and set the location to be **Horizontally** right and **Vertically** top.
  c. Make the text bold using the B **Bold** icon.
  d. Drag a Value Prompt **Value Prompt** object into the second column on your prompt page. This will bring up the Prompt Wizard.
3. On the **Choose Parameter** screen of the Prompt Wizard:
  a. Select the option **Create a new parameter**.
  b. In the box, type a name for the parameter. For example, **Operating Effectiveness:**.
  c. Click **Next**.
4. On the **Create Filter** screen:
  a. Select the option **Create a parameterized filter**.
  b. In the **Model item** box, click the browse button and navigate to the following query item:

```
[DEFAULT_REL]|ENUMERATIONS|SOXCONTROL(ENUMERATIONS)|
[CN_OPERATING_EFFECTIVENESS]|[ENUMVALID]
```

c. In the **Operator** box, use = (the equal sign is the default).

d. Clear the box for the **Make this filter optional** option.

e. If you have multiple queries in your report, in the **Apply filter** box, select the query to which the filter should be applied.

f. Click **Next**.

5. On the **Populate Control** screen:

   a. Select the **Create new query** option.

   b. In the **Name** box, type a name for the query. For example, **Operating Effectiveness Prompt** Query.

   c. In the **Values to display** box, click the browse button and navigate to the `following` query item:

   ```
   [DEFAULT_REL]|ENUMERATIONS|SOXCONTROL(ENUMERATIONS)|
   [CN_OPERATING_EFFECTIVENES]|[CN_OPERATING_EFFECTIVENES]
   ```

   d. The following values will be displayed in the box:

   ```
   [DEFAULT_REL].[CN_OPERATING_EFFECTIVENES].[CN_OPERATING_EFFECTIVENES]
   ```

   e. Click **Finish**.

6. Now select the prompt you just created and under the **General** category in the Properties pane, change the **Select UI** property to **List Box**.

7. If there are a limited number of items in the prompt, you can set the prompt height to a value that shows just the prompts and no white space:

   a. Select the prompt in the table that you want to set the height.

   b. In the Properties pane, under **Positioning**, select **Size and Overflow**.

   c. In the dialog box, set the height to 14 pt times the number of rows in the prompt.

   d. Click **OK**.

8. The filter produced in the selected query of the report will appear as follows:

   ```
   [DEFAULT_REL].[CN_OPERATING_EFFECTIVENESS].[ENUMVALID] = ?Operating
   Effectiveness?
   ```

   a. The problem with creating the above filter is that is causes Oracle to prefer the index associated with enumerated string over a better performing system index to join tables related to the objects above and below this object. In order to solve this problem modify the filter corresponding to this prompt by adding a **+ 0** (plus zero) to the left side of the filter expression to disable the use of indexes on this field.

   ```
   [DEFAULT_REL].[CN_OPERATING_EFFECTIVENESS].[ENUMVALID] + 0 = ?Operating
   Effectiveness?
   ```

9. Run the report.

Results

For details and more explanation, see the section on Chapter 5, “Optimizing Performance,” on page 39.

Adding a Multi-select Enumerated Value Prompt

Follow this procedure to add a multi-select enumerated value prompt to your report.

Procedure

1. On the prompt page, select the value prompt you just created.

2. In the **Properties Pane** under the **General** category, set the **Multi-Select** property value to **Yes**.
3. In the **Query Explorer**, select the query where the filter is located.
4. Select the filter that is related to this prompt. Change the operator from = to **in**. Parentheses are not required.
5. Click **OK**.
6. Run the report.

Results

You can now select two values in the prompt in addition to having **Select all** and **Deselect all** links for the users to access.

Adding an All Static Value to a Multi-select Enumerated Value Prompt

There are instances where the customer has hidden the Select all and Deselect all links but still wants their users to have the ability to Select All as a static choice.

Follow this procedure to add a Select All static choice to your report.

Procedure

1. On the prompt page, select the value prompt you just created.
2. In the **Properties Pane** under the **Data** category, click on the **Static Choices** property.
   a. Click the Add icon.
   b. In the Edit dialog, set the **Use** value to -1 and **Display** value to All.
   c. Click **OK**.
   d. Click **OK**.
3. We will also set the All value to be the default when the prompt page appears. Do the following:
   a. In the Properties pane under the **General** category, click on **Default Selections**.
   b. Click the Add icon.
   c. Enter **-1** for the value.
   d. Click **OK**.
   e. Click **OK**.
4. In the Properties pane, under **Positioning**, select **Size and Overflow**.
   a. In the dialog box, set the height to **56 pt**.
5. In the filter for the multi-select prompt, you must now go and add `?Operating Effectiveness? = -1` to the filter.
6. Run the report.

Results

This fixes the prompt itself, now we have to fix the filter. The filter currently looks as follows:

In order to account for the static value of "All" that you just added, change the filter as follows:

```
[DEFAULT_REL].[SOXCONTROL].[CN_OPERATING_EFFECTIVENESS_ID] in
?Operating Effectiveness? or ?Operating Effectiveness? = -1
```

This bypasses the filter against the database and nullifies the use of the filter so all values are accepted.

Adding a Date Prompt

Follow this procedure to add a date prompt to your report.

Procedure

1. If you do not already have a prompt page in your report, follow the instructions in "Creating a Prompt Page" on page 54.
2. To create your Date prompt, do the following:
   a. Drag a ab Text Item **Text Item** into the first column of the table. Enter the text for the prompt in the text item.
   b. Select the table cell and set the location to be ≣ horizontally right and ▭ vertically top.
   c. Make the text bold using the **B** bold icon.
   d. Drag a Date Prompt **Date Prompt** object into the second column of your prompt page. This will bring up the Prompt Wizard.
3. On the **Choose Parameter** screen of the Prompt Wizard:
   a. Select the option **Create a new parameter**.
   b. In the box, type a name for the parameter. For example, **Start Date:**.
   c. Click **Next**.
4. On the **Create Filter** screen:
   a. Select the option **Create a parameterized filter**.
   b. In the **Model item** box, click the browse button and navigate to the following query item:
      `DEFAULT_REL|GRC_OBJECTS|[SOXCONTROL]|[CN_CREATION_DATE]`
   c. In the **Operator** box, use **=** (the equal sign is the default).
   d. Clear the box for the **Make this filter optional** option.
   e. If you have multiple queries in your report, in the **Apply filter** box, select the query to which the filter should be applied.
   f. Click **Next**.
5. Now select the prompt you just created and under the **General** category in the Properties pane, change the **Select UI** value to **Edit Box**.
6. Run the report.

Adding a Date Range Prompt

The process for creating a date range prompt requires that you add two Date Prompt objects on the prompt page (one for a beginning date and the other for an ending date) of your report, and modify a filter for the ending date.

For example if the user selects a start date of Jul 1st, 2006 and an end date of Jul 7th, 2006 from a date range prompt, effectively the filter will be applied between Jul 1st 2006 00:00 and Jul 7th, 2006 00:00. To retrieve the records irrespective of the time of the end date, you need to modify the filter.

In the following example, a filter was created for the query item `[CREATED_ON]` in the query subject `[ACTOR_AUDIT_TRAIL]`.

Follow this procedure to create a date range prompt and modify the filter for the end date.

Procedure

1. Create a date prompt for the start date as described in "Adding a Date Prompt" on page 60. Use the parameter name of **Start Date**. In the Prompt Wizard, change the operator to >=.
2. Similarly create a date prompt for the end date as described in "Adding a Date Prompt" on page 60. Use the parameter name of **End Date**. In the Prompt Wizard change the operator to <=.
3. In the Detail Filters pane for the query, edit the right side of the filter expression for the query item by adding the function `_add_days(?End Date?,1)` after the <= operator. This will make sure the date range includes the entire end date.

Results

For example, `[OPENPAGES_REL].[ACTOR_AUDIT_TRAIL].[CREATED_ON] <= _add_days(?EndDate?,1)`. The function `_add_days` adds one more day to the end date.

Adding a Static Choices Prompt

In some cases, a prompt cannot be populated with values based on a query item. If, for example, a report contains two columns: Issue Status and Issue Date, and you want the report to sort either by the Issue Status column or the Issue Date column depending on the user selection, then you would have to create a value prompt that would use static choices (such as a column heading).

Follow this procedure to add a value prompt with static choices that can be used in filters for a report.

Procedure

1. If you do not already have a prompt page in your report, follow the instructions in "Creating a Prompt Page" on page 54.
2. To create your Static prompt, do the following:
   a. Drag a ab Text Item **Text Item** into the first column of the table. Enter the text for the prompt in the text item.
   b. Select the table cell and set the location to be ≣ **Horizontally** right and ▭ **Vertically** top.
   c. Make the text bold using the **B** **Bold** icon.
   d. Drag a ▾ Value Prompt **Value Prompt** object into the second column of your prompt page. This will bring up the Prompt Wizard.
3. On the **Choose Parameter** screen of the Prompt Wizard:
   a. Select the option **Create a new parameter**.
   b. In the box, type a name for the parameter. For example, **Sort Order**.
   c. Click **Finish**.
4. Now you can add the static values to the prompt:
   a. Select the newly created prompt object.
   b. In the **Properties** box under **Data**, click the **Static Choices** button to open the **Static Choices** box.

5. In the **Static Choices** box:
   a. Click the **Add** button to open the **Edit** box.
   b. In the **Use** field, enter the value to use in the filter. For example, 1.
   c. In the **Display** field, enter the value to be displayed in the prompt. For example, Issue Status.
   d. To add more values, repeat Steps a-c.

Results

You can use the newly created parameter, for example Sort order, to filter your reports. For information on sorting data, see "Sorting Data" on page 69.

Adding a Cascading Prompt

Cascading prompts are prompts that are dynamically populated based on the choice made in the previous prompt. For example, if you choose a Reporting Period in a prompt, the second prompt might be populated with the Business Entities that exist in that Reporting Period.

Follow this procedure to add a cascading prompt to an Entity prompt for a selected reporting period.

Procedure

1. If you do not already have a prompt page in your report, follow the instructions in "Creating a Prompt Page" on page 54.
2. If you have not already created a reporting period prompt, follow the instructions in "Adding a Reporting Period Prompt" on page 55.
3. If you have not already created a business entity prompt, follow the instructions in "Adding a Business Entity Prompt" on page 56.
4. Select the **Reporting Period** prompt.
5. In the Properties pane of the **Reporting Period** prompt under **General**, do the following:
   a. Set the **Auto-Submit** property to **Yes**.
   b. If the **Default Selections** property contains a default value, remove it.
6. Select the **Business Entity** Prompt.
7. In the Properties pane of the **Business Entity** prompt, do the following:
   a. Set the **Cascade Source** property to the Reporting Period prompt's parameter **Reporting Period**. This will enable the cascading prompt on the Entity Prompt.
8. In the **Business Entity Prompt** query, do the following:
   a. Make sure there is a reporting period query item.
   b. Create a filter using the reporting period query item and the ?Reporting Period? prompt.

Results

The business entity prompt is not populated because it is waiting for the value to be entered in the reporting period prompt. Since you set the Auto-Submit value to Yes, as soon as the user selects a value in the Reporting Period prompt, the business entity prompt query will be executed with the reporting period filter.

Adding Prompts for Query Subjects

Within the IBM OpenPages GRC on Cloud framework model, some query subjects contain prompt macros. When that query subject is used in a report, you have to provide values for those prompts (for prompt macros and values, see Chapter 4, “Reporting Framework,” on page 15). For example, the query subject [MODIFIED RESOURCES BY ENTITY] (located in the DEFAULT|[DEFAULT_REL]|AUDIT_TRAIL folder) contains prompt macros. When you use that query subject in a report, a prompt page is generated.

The prompt page generated is not user friendly. To assist the report user, you can create your own report prompt page using the names of the existing prompt macros. For example, you can create a value prompt for Business Entity using the existing parameter **parent entity ID** as the output prompt parameter. When you now run the report, first the report prompt page appears and then the self-generated prompt page appears with the remaining parameters.

You can eliminate the self generated prompt page by creating prompts for all of prompt macros in the query.

The methods for creating these prompts is identical to those already described in this chapter. The only difference is that in the prompt wizard on the **Choose Parameter** screen, you select the **Use existing parameter** option and select the appropriate prompt from the list.

Adding Rich Text Fields

If the text is not HTML or rich text, it will be displayed as plain text in the report.

Text fields in IBM OpenPages GRC on Cloud can be set to display rich text. Special formatting must take place in reports to display these fields correctly.

Follow this procedure to use a rich text field.

Procedure

1. Open a list report.
2. From the **Insertable Objects** pane, drag and drop the following data items into the list:
   - `DEFAULT|[DEFAULT_REL]|GRC_OBJECTS|SOXBUSENTITY_FOLDER|[SOXBUSENTITY_GPC]|[CEN_NAME00]`
   - `DEFAULT|[DEFAULT_REL]|GRC_OBJECTS|[SOXPROCESS]|[PR_NAME00]`
   - `DEFAULT|[DEFAULT_REL]|GRC_OBJECTS|[SOXPROCESS]|[PR_DESCRIPTION]`
   - `DEFAULT|[DEFAULT_REL]|GRC_OBJECTS|[SOXPROCESS]|[PR_ADDITIONAL_DESCRIPTION] (This is defined as a Rich Text field)`
3. Click the **Toolbox** tab in the **Insertable Objects** pane, and then drag and drop an **HTML Item** object onto the list.
4. Select the HTML item in the list:
   a. In the Properties pane under **HTML Source**, set the **Source Type** property to **Data Item Value**.
   b. Set the **Data Item Value** property to [PR_ADDITIONAL_DESCRIPTION].
5. Cut the [PR_ADDITIONAL_DESCRIPTION] from the list.
6. Select the HTML Item column header:

   a. In the **Properties** pane under **Text Source**, set the **Source Type** property to **Data Item Label**.
   b. Set the **Data Item Label** property to [PR_ADDITIONAL_DESCRIPTION].
   c. Run the report.

   This formats correctly when you display the report in HTML but when you render the report in either a PDF or Excel format, the rich text displays as a blank field. In order to correct this, do the following:
7. In the **Query Explorer**, open the list query.
8. Add a new **Data Item** to the query:
   a. Drag a **Data Item** into the data items pane.
   b. Enter the following in the Expression Definition pane:
      ```
      trim(regexp_replace([DEFAULT_REL].[SOXPROCESS].[PR_ADDITIONAL_DESCRIPTION],
      '<[^>]+>'))
      ```
   c. **Validate** the expression, and then click **OK**.
9. In the Properties pane of the data item you just added:
   a. Under the **Data** group, set the **Name** property to **Additional Description**.
   b. Under the **Data** group, set the **Aggregate Function** property to **None**.
   c. Under the **Data** group, set the **Rollup Aggregate Function** property to **None**.
10. Go to the main report page.
11. Drag and drop **Additional Description** into the list.
12. In the Properties pane of list column title of **Additional Description**, do the following:
    a. Under T**ext Source**, set the **Source Type** property to **Data Item Label**.
    b. Set the **Data Item Label** to **PR_ADDITIONAL_DESCRIPTION**.
13. Now we need to create a conditional variable. Do the following:
    a. In the **Condition Explorer** click on **Variables**.
    b. Drag and drop a **String Variable** into the Variables pane.
    c. In the Expression Definition pane, enter the following: ReportOutput () and click **OK**.
    d. Using the **Add** icon, enter **HTML** as a variable.
    e. In the Properties pane under **Miscellaneous**, in the **Name** property enter **Report Output**.
14. Return to main report page and do the following:
    a. Select the column with the **HTML item**.
    b. Using the ancestor selector, select **List Column**.
    c. In the Properties pane under **Conditional**, set the **Style Variable** property to **Report Output**.
    d. In the **Condition Explorer** select **Other** under the **Report Output** variable.
    e. In the Properties pane under **Box**, set the **Box Type** property to **None**.
    f. In the **Condition Explorer** select **(no variables)**.
15. Now do the following:
    a. Select the column with the **<Additional Description>** in the list body.
    b. Using the ancestor selector, select **List Column**.
    c. In the Properties pane under **Conditional**, set the **Style Variable** property to **Report Output**.
    d. In the **Condition Explorer** select **HTML** under the **Report Output** variable.

   e. In the Properties pane under **Box**, set the **Box Type** property to **None**.
   f. In the **Condition Explorer** select **(no variables)**.

16. Run the report.

Results

The report will look the same as before when run in HTML format. However, now when you run the report as a PDF, the rich text will appear as plain text in the report.

The conditional formatting we created shows either the HTML formatted column or the other Additional Description column when HTML is not the output format.

In this example we used an HTML object to display rich text. There is a Rich Text object in the toolbox, but it only supports an extremely limited amount of rich text properties so we do not recommend using it.

Adding CrossTrack Links

Follow this procedure to create CrossTrack links in a report from a Control object to its details page.

A CrossTrack link is a hyperlink in the report that when clicked, takes you to a detail page in IBM OpenPages GRC on Cloud.

Procedure

1. Create a new list report.
2. From the **Query Explorer**, open the query.
   a. In the Properties pane under **Miscellaneous**, set the name to **listMain**.
3. From the **Page Explorer**, select the main report page.
   a. Click the **Source** tab in the Insertable Objects pane.
   b. Drag the following query items into the List object box:
      - `DEFAULT|[DEFAULT_REL]|GRC_OBJECTS|[SOXCONTROL]|[CN_NAME00]`
      - `DEFAULT|[DEFAULT_REL]|GRC_OBJECTS|[SOXCONTROL]|[CN_DESCRIPTION]`
      - `DEFAULT|[DEFAULT_REL]|GRC_OBJECTS|[SOXCONTROL]|ENUMERATION_FIELDS| CLASSIFICATION (ENUMERATION)|[CN_CLASSIFICATION]`
      - `DEFAULT|[DEFAULT_REL]|GRC_OBJECTS|[SOXCONTROL]|ENUMERATION_FIELDS| DESIGN_EFFECTIVENESS (ENUMERATION)|[CN_DESIGN_EFFECTIVENESS]`
      - `DEFAULT|[DEFAULT_REL]|GRC_OBJECTS|[SOXCONTROL]|[CN_DETAIL_PAGE_URL]`
4. Select the `[CN_DETAIL_PAGE_URL]` column in the report and cut it (not delete it).
5. Change the list column titles:
   a. Select each list column title.
   b. In the Properties pane under **Text Source**, set the **Source Type** property to **Text**.
   c. Double-click in the **Text** property and type in the appropriate title value. Set the headers to the following values:
      - `[CN_NAME00]` to `Control Name`.
      - `[CN_DESCRIPTION]` to `Description`.
      - `[CN_CLASSIFICATION]` to `Classification`.
      - `[CN_DESIGN_EFFECTIVENESS]` to `Design Effectiveness`.
6. Add the following JavaScript to the report:

Drag an HTML Item **HTML** item to the top of the report.

Double-click on the **HTML** item and enter the following code:

```
<script language="javascript">
var _crosstrackTargetWindow;
function _discoverCrosstrackTargetWindow()
{
    if(typeof(_crosstrackTargetWindow)=='undefined')
    {
        var win;
        if(typeof(window.opener)=='undefined')
        {
            win = window.parent;
        }
        else if(typeof(window.opener.opener)=='undefined')
        {
            if(typeof(window.opener.parent)=='undefined')
            {
                win = window.opener;
            }
            else
            {
                win = window.opener.parent;
            }
        }
        else
        {
            win = window.opener.opener;
        }
        _crosstrackTargetWindow = win;
    }
}
_discoverCrosstrackTargetWindow();
function _crosstrack(url, targetWin)
{
    if(typeof(targetWin)=='undefined')
    {
        if(typeof(window.opener)=='undefined')
        {
            targetWin = window.parent;
        }
        else if(typeof(window.opener.opener)=='undefined')
        {
            if(typeof(window.opener.parent)=='undefined')
            {
                targetWin = window.opener;
            }
            else
            {
                targetWin = window.opener.parent;
            }
        }
        else
        {
            targetWin = window.opener.opener;
        }
    }
    targetWin.location.href=url;
    targetWin.focus();
}
function crosstrack(url)
{
    _crosstrack(url, _crosstrackTargetWindow);
}
</script>
```

7. Click the **Toolbox** tab, and then drag a **Hyperlink** object into the first column of the list report.
8. From the toolbar, click the **Padlock** button to unlock the template (the icon changes to unlocked).
9. Now you need to make the hyperlink data item a property of the list.
   a. Select the **List** object from the main report page.
   b. In the Properties pane under the **Data** category, click **Properties**.
   c. Select `[CN_DETAIL_PAGE_URL]`.
   d. Click **OK**.
10. Configure the **Hyperlink** object as follows:
    a. Click the **Hyperlink** text in the cells of the **Hyperlink** object column.
    b. In the Properties pane under the **Text Source** category, set the **Source Type** property to **Data Item Value** and the **Data Item Value** property to `[CN_NAME00]`.
    c. In the Properties pane under the **URL Source** category, set the **Source Type** property to **Report Expression**.
    d. Click in the **Report Expression** property and enter the following JavaScript code in the Expression Definition pane:

    ```
    if (ReportOutput()="HTML")
    then
    ("javascript:crosstrack('"+ [listMain].[CN_DETAIL_PAGE_URL] + "');")
    else
    ([listMain].[CN_DETAIL_PAGE_URL])
    ```

    Note: The items in italics are the values that need to be replaced when you implement this on your system.

    This code causes the hyperlink to be opened in the existing OpenPages GRC on Cloud window.

    e. Select the column header text in the **Hyperlink** object.
    f. In the Properties pane under the **Text Source** category, set the **Text** property to **Control Name**.
11. From the **Toolbar**, click the **Padlock** button again to lock the template (the icon changes to locked).
12. Select the original `[CN_NAME00]` column (not the hyperlink column) on the report page and then cut it.
13. Run the report.

    Note: In order to test the CrossTrack link, the report must be published and run from OpenPages GRC on Cloud.

Results

When a user clicks the CrossTrack link in the report, the detail page opens in the application window from which the report was launched.

When you click on the hyperlink, you will be taken to the detail page for that object.

Adding Images and Icons

You can enhance the look of your reports by adding icons or a company logo to your reports.

The image files are typically located in the following directory on the IBM Cognos server:

```
C:\OpenPages\Cognos\cognos\c8\webcontent\skins\branding\<icon file name>
```

It is better to use a relative path:

```
../skins/branding/<icon file name>
```

Important: Please talk with your system administrator about identifying the correct image directory path for your particular installation. These paths are different for AIX®, Linux, and the Microsoft Windows operating system.

Follow this procedure to add images to your report.

Procedure

1. Create a new list report.
2. Click the **Toolbox** tab, and then drag a **Image** object into the first column of the list report.
3. From the toolbar, click the **Padlock** button to unlock the template (the icon changes to unlocked).
4. Select the **Image** object.
5. In the Properties pane of the **Image** object:
   a. Under the **URL Source** category, click on the **URL** property.
   b. In the Image URL field, enter the following: ../skins/branding/icon-SOXControl.gif to add the control icon to the report. You must use a relative path as shown in this example or the report will not be portable between systems. You must also use the correct capitalization for the name of the icon or they will not appear correctly on an AIX or Linux system.
   c. Click **OK**.
6. From the **toolbar**, click the **Padlock** button again to lock the template (the icon changes to locked).
7. Run the report.
8. The image now appears as follows in your list.

Results

Images can all be used as a CrossTrack link in the same manner that the hyperlink object was used. Images can also be dropped into a hyperlink so the user can click the image or the hyperlink and obtain the same behavior.

Adding Calculations

There are two methods for creating calculations in the framework.

- You can create computed fields through IBM OpenPages GRC on Cloud (“Creating a Computed Field” on page 210).
- You can create a calculation in the custom query subject file (“Creating Calculations” on page 228).

Once the calculations are created and the framework is regenerated, they become available to the report author in the computed fields folders in each namespace.

Calculations can be dropped into any report just like any other query item. They can be aggregated using the method described in “Aggregating Data” on page 75.

Sorting Data

You can organize data in a report column by specifying a sort order. Data can be organized in ascending or descending order based on the values in any column.

Using Simple Sorting

Follow this procedure to sort a column using the built-in sorting function.

Procedure

1. Open a report.
2. Click the column on which you want to sort.
3. Click the **Sorting** icon on the toolbar, and then click **Sort Ascending** or **Sort Descending**.
4. Run the report.

Using Advanced Sorting

The advanced sorting function is used to apply sorts on multiple columns in a report. It enables the report author to specify the column order and sort direction that they desire the report to be sorted by.

Procedure

1. Open a report.
2. Click the **Sorting** icon on the toolbar, and then click **Advanced Sorting**.
3. In the Advanced Sorting dialog, determine which columns you wish to sort by and drag and drop them under the **Detail Sort List** folder.
4. You can change the order with the up/down arrows or change the sort direction.

Results

If there were groups created in this list, you would see them in the groups folder. Sorting of groups takes place before detail sorting of the list. The advanced sorting dialog gives you the power to create a sort that is not based upon the order in which columns appear in the report.

Using Custom Sorting

There may be situations in which either sorting data in an ascending or descending order does not meet your requirements.

For example, consider a report that shows control names and their operating effectiveness. The "Operating Effectiveness" field can have one of these values: "Effective", "Ineffective" or "Not Determined."

If you want to show the control values for the "Operating Effectiveness" field in this order: "Ineffective," "Not Determined," and "Effective," you cannot achieve this result by sorting the "Operating Effectiveness" column in ascending or descending order. If the column is sorted by ascending order, then the values in the report will display as "Effective", "Ineffective" and "Not Determined." If the column is sorted by descending order, then the values will display as "Not Determined", "Ineffective" and "Effective."

To achieve a custom sort order, you have to create a new data item in the underlying **Query Subject** and then sort that data item.

The following example demonstrates how to create a custom sort order for the control object's "Operating Effectiveness" field in a new list report.

Procedure

1. Create a new list report with the following query items:
   - `DEFAULT|[DEFAULT_REL]|GRC_OBJECTS|[SOXCONTROL]|[CN_NAME00]`
   - `DEFAULT|[DEFAULT_REL]|GRC_OBJECTS|[SOXCONTROL]| ENUMERATION_FIELDS| OPERATING_EFFECTIVENES (ENUMERATION)| [CN_OPERATING_EFFECTIVENES]`
   - `DEFAULT|[DEFAULT_REL]|GRC_OBJECTS|[SOXCONTROL]| ENUMERATION_FIELDS| OPERATING_EFFECTIVENES (ENUMERATION)| [CN_OPERATING_EFFECTIVENES_ID]`
2. Run the report and observe the order of controls.
3. Using the **Query Explorer** select the query you are going to custom sort.
4. Click the **Toolbox** tab in the **Insertable Objects** pane, and then drag and drop a **Data Item** object onto the **Data Items** box of the query.
5. In the Expression Definition pane enter the following expression:

```
Case ([CN_OPERATING_EFFECTIVENES_ID] + 0)
when #$SOXCONTROL_OPERATING_EFFECTIVENES_DEFINITION_MAP
    {"Ineffective"}# then 1
when #$SOXCONTROL_OPERATING_EFFECTIVENES_DEFINITION_MAP
    {"Not Determined"}# then 2
when #$SOXCONTROL_OPERATING_EFFECTIVENES_DEFINITION_MAP
    {"Effective"}# then 3
else 4
end
```

6. Validate the expression, and then click **OK**.
7. In the Properties pane of the data item:
   a. Under **Data**, set the **Name** property by typing a name for the data item. For example, **Sort Order**.
   b. Under Data set the **Aggregate Function** property to **None**.
   c. Under Data set the **Rollup Aggregate Function** property to **None**.
8. Following the instructions under "Using Advanced Sorting" on page 69, add the **Sort Order** data item to the **Detail Sort List**.
9. Run the report.

Using Prompted Sorting

Follow this procedure to sort a report on the column selected by the user from the report's prompt page.

Procedure

1. Create a new list report with the following query items:
   - `DEFAULT|[DEFAULT_REL]|GRC_OBJECTS|[SOXCONTROL]|[CN_NAME00]`
   - `DEFAULT|[DEFAULT_REL]|GRC_OBJECTS|[SOXCONTROL]| ENUMERATION_FIELDS| OPERATING_EFFECTIVENESS (ENUMERATION)|[CN_OPERATING_EFFECTIVENES]`
   - `DEFAULT|[DEFAULT_REL]|GRC_OBJECTS|[SOXCONTROL]| ENUMERATION_FIELDS| OPERATING_EFFECTIVENESS (ENUMERATION)|[CN_DESIGN_EFFECTIVENES]`
2. Create a prompt page.

3. Add a **Value Prompt** with static choices to the prompt page. For details on how to add a value prompt with static choices, see “Adding a Static Choices Prompt” on page 61. Name the prompt **Sort Order**.
4. In the **Properties** box of the **Value Prompt**, add the following values in the **Static Choices** box:

In the "Use" field enter this...	In the "Display" field enter this...
1	Operating Effectiveness
2	Design Effectiveness

5. Using the **Query Explorer** select the query you are going to edit.
6. From the Tabular Model:
   a. Click the **Toolbox** tab and drag a **Data Item** object into the Data Items pane.
   b. In the Expression Definition pane, enter the following:

      ```
      case ?Sort Order?
      when 1 then [CN_OPERATING_EFFECTIVENES]
      when 2 then [CN_DESIGN_EFFECTIVENESS]
      end
      ```

      Where:

      Sort Order is the name of the parameter created in step 3 of this example.

      The values ‘1’ and ‘2’ in the above expression are the **Use** values that were created in Step 4. If the values are different from the ones shown here, you have to change the above expression.
7. In the Properties pane of the Data Item, do the following:
   a. Change the name of the data item to **Sort Order**.
   b. Change the **Aggregate Function** property to **None**.
   c. Change the **Rollup Aggregate Function** property to **None**.
8. Following the instructions under “Using Advanced Sorting” on page 69, add the **Sort Order** data item to the **Detail Sort List**.
9. Run the report. Depending on the user selection in the prompt, the report will be sorted.

Filtering Data

You can use a report filter to focus a report and minimize processing time by excluding unwanted data.

Using Parameter Maps for Filtering

In order to display, for example, on a report only those control objects that have an operating effectiveness value of "Effective", you would apply a filter on the "Operating Effectiveness" enumerated string field.

The data type of the "Operating Effectiveness" field is an enumerated (enum) data type. As the IBM OpenPages GRC on Cloud application is fully internationalized, the enumerated values are translated into the locale of the user who logged into the application.

If you were to use the string value of "Effective" in a filter expression, such as:

```
[CN_OPERATING_EFFECTIVENES] = 'Effective'
```

The enumerated value would work only for the US English locale, it will not work for other locales. For example, if a user whose locale is Japanese logs in, the enumerated values will be translated to Japanese and so the above filter definition will not work.

For this filter to work in any locale, you must use the ID of the enumerated string value in the filter expression instead of the enumerated string value itself.

The query subject for the corresponding object has two query items, one that contains the ID information for the enumerated string value and the other that contains the actual enumerated string value.

For example, in the generated framework model, the SOXCONTROL query subject contains two fields:[CN_OPERATING_EFFECTIVENES_ID] and [CN_OPERATING_EFFECTIVENES].

To filter on the enumerated string value "Effective" so that it works in all locales, you would have to modify the previous expression to use the ID of the enumerated string value, as follows:

```
[CN_OPERATING_EFFECTIVENES_ID] = 45
```

where 45 is the ID of the enumerated string value "Effective" for the "Operating Effectiveness" object field in this particular data set. However, the ID of the enumerated string can change from one deployment to another. In practice, the ID must be obtained through either a prompt page or a parameter map.

Parameter maps are created for all single valued and multi-valued enumerated fields.

You can use parameter maps in the filter expression instead of finding the IDs of the enumerated string values. For example:

```
[CN_OPERATING_EFFECTIVENES_ID] + 0 =
#$SOXCONTROL_OPERATING_EFFECTIVENES_DEFINITION_MAP{"Effective"}#
```

When the report is run, the parameter map definition is replaced by the actual ID of the value "Effective".

The naming convention for parameter maps is as follows:

```
#$<Object Name>_<Field Name>_DEFINITION_MAP{"Value"}#
```

Where <Object Name> is the name of the object and <Field Name> is the name of the enumerated field.

Filtering on Reporting Period ID

Every report should be filtered by the reporting period ID (either through the use of a prompt or a filter) unless the user wants to report across all reporting periods.

You could, for example, prompt the user explicitly for a specific reporting period and filter by the selection. Or, you could filter by the Reporting Period ID value -1 if the report is only available for the current reporting period.

Tip: When combining filtering on Reporting Period ID with filtering on entity, you must use the GEN_REPORTING_PERIOD_ID as the filtering item for the Reporting Period ID.

Filtering on an Enumeration

There are two types of enumerated variables, single valued and multivalued. In order to filter using enumerations, the operator of the filter must change to accommodate the type of prompt that is being used.

Table 42. Filter Operators

Enumeration Type	Prompt Type	Filter Operator	Comment
Single Value	Single select	=	
Single Value	Multi-select	in	
Multivalued	Single select	contains	Returns any record that contains the value in the prompt.
Multivalued	Multi-select	*in	Returns any records that contain any of the selected values in the prompt.

In order to filter a multivalued enumerated string against a multi-select prompt you must use the following syntax for the filter:

```
[DEFAULT_REL].[RI_DOMAIN].[RI_DOMAIN] in ?Domain?
```

This filter is using the <object_field name> data item from the definition of the domain (as an example) enumerated string to be compared against the prompt. This field returns individual records for each of the domains, that the user could have selected in the application, to be compared against the prompt value which could also be multiple selections.

In addition you **MUST** make sure the **Auto Group & Summarize** property on the query subject which contains this filter is set to **Yes**.

Grouping Data

You can remove duplicate values in a report by grouping query items. For example, if a report contains multiple controls for a particular risk and business entity, you can group the risk and business entity columns so only one instance of the risk and business entity is displayed in the list for each control.

Grouping Data in a List Report

Follow this procedure to group data in a new list report. The sample report uses the business entity, risk, and control query items and then groups the business entity and risk columns to remove duplicate data.

Note: You should group by Resource ID as opposed to Name because Resource IDs are unique throughout the system when names are not.

Procedure

1. Create a new list report with the following query items from the relational namespace placed in the **Data Items** pane of the query:
   - `DEFAULT|[DEFAULT_REL]|GRC_OBJECTS|SOXBUSENTITY_FOLDER| [SOXBUSENTITY_GPC]|ID_FIELDS|[CEN_ENTITY_ID]`
   - `DEFAULT|[DEFAULT_REL]|GRC_OBJECTS|SOXBUSENTITY_FOLDER| [SOXBUSENTITY_GPC]|[CEN_NAME00]`
   - `DEFAULT|[DEFAULT_REL]|GRC_OBJECTS|[SOXRISK]| ID_FIELDS|[RI_RISK_ID]`

- DEFAULT|[DEFAULT_REL]|GRC_OBJECTS|[SOXRISK]|[RI_NAME00]
- DEFAULT|[DEFAULT_REL]|GRC_OBJECTS|[SOXCONTROL]|
  ID_FIELDS| [CN_CONTROL_ID]
- DEFAULT|[DEFAULT_REL]|GRC_OBJECTS|[SOXCONTROL]|[CN_NAME00]

2. Group the data for the following columns:
   a. Select the [CEN_ENTITY_ID] column.
   b. Click the Group icon. The group icon will be displayed in the column you just selected.
   c. Select the [RI_RISK_ID] and [CN_CONTROL_ID] column, and repeat Step b.
3. Cut the following query items from the list to hide the object's identifier from appearing on the report page:

   ```
   [CEN_ENTITY_ID]
   [RI_RISK_ID]
   [CN_CONTROL_ID]
   ```

   Important: When you cut a column from a report it will not be shown in the report, but it will still be present in the underlying query. When you delete a column from a report, it will be deleted from the report and the underlying query if no other objects in the report reference that query item.
4. Now select the [CEN_NAME00] column and in the **Properties Pane**, under **Data**, select **Group Span** and set it to [CEN_ENTITY_ID].
5. Now select the [RI_NAME00] column and in the **Properties Pane**, under **Data**, select **Group Span** and set it to [RI_RISK_ID].
6. Now select the [CN_NAME00] column and in the **Properties Pane**, under **Data**, select **Group Span** and set it to [CN_CONTROL_ID].
7. Now select the [CN_DESCRIPTION] column and in the **Properties Pane**, under **Data**, select **Group Span** and set it to [CN_CONTROL_ID].
8. Run the report.

Creating a List Group Header

Follow this procedure to use the name of an object (such as the name of an entity) as a group header in a new list report.

The sample report uses the business entity, risk, and control query items. It then groups the business entity and risk columns to remove duplicate data, and displays the name of the entity from the query subject [CEN_NAME00] as a header in the sample report.

Do the following:

Procedure

1. Follow Steps 1-3 as described in "Grouping Data in a List Report" on page 73 to create a list report with groupings.
2. From the **Structure** menu, click the **Headers & Footers | List Headers & Footers...** menu item to open the **List Headers & Footers** box:
   a. Select CEN_NAME00(header).
   b. Click **OK**.
3. On the report page, cut the [CEN_NAME00] query item from the list to hide the object's identifier from appearing as a list column in the report.
4. Run the report.

Setting the Level Span for a Column

When columns are grouped, you can choose how often to show the column values by using level spanning. Unlike grouping, level spanning will not change the underlying query.

Procedure

1. Create a new list report with the following query items from the relational namespace:
   - `DEFAULT|[DEFAULT_REL]|GRC_OBJECTS|[SOXRISK]|ID_FIELDS|[RI_RISK_ID]`
   - `DEFAULT|[DEFAULT_REL]|GRC_OBJECTS|[SOXRISK]|[RI_NAME00]`
   - `DEFAULT|[DEFAULT_REL]|GRC_OBJECTS|[SOXRISK]| ENUMERATION_FIELDS| INHERENT_IMPACT (ENUMERATION)| [RI_INHERENT_IMPACT]`
   - `DEFAULT|[DEFAULT_REL]|GRC_OBJECTS|[SOXRISK]| ENUMERATION_FIELDS| INHERENT_LIKELIHOOD(ENUMERATION)| [RI_INHERENT_LIKELIHOOD]`
   - `DEFAULT|[DEFAULT_REL]|GRC_OBJECTS|[SOXCONTROL]|ID_FIELDS| [CN_CONTROL_ID]`
   - `DEFAULT|[DEFAULT_REL]|GRC_OBJECTS|[SOXCONTROL]|[CN_NAME00]`
   - `DEFAULT|[DEFAULT_REL]|GRC_OBJECTS|[SOXCONTROL]|ENUMERATION_FIELDS| OPERATING_EFFECTIVENES(ENUMERATION)|[CN_OPERATING_EFFECTIVENES]`
2. Modify the report page as follows:
   a. Cut the following query items from the list to hide the object's identifier from appearing on the report page:

      ```
      [RI_RISK_ID]
      [CN_CONTROL_ID]
      ```

   b. Group the data on the [RI_NAME00] column.
3. Run the report.

   [RI_NAME00] is not repeated for all of its controls. However, [RI_INHERENT_IMPACT], and [RI_INHERENT_LIKELIHOOD], which are the properties of the risk object, are repeated for each associated control object in the generated report.

   In order to have the additional risk fields not repeat, do the following:
4. To level span the [RI_INHERENT_IMPACT], and [RI_INHERENT_LIKELIHOOD] query items when [RI_NAME00] changes:
   a. Select the [RI_INHERENT_IMPACT] column in the report.

   b. In the **Properties** box under **Data** for the selected column, click the **Group Span** button and select the [RI_NAME00] column.

   c. Repeat Steps a and b for the [RI_INHERENT_LIKELIHOOD] column.

Results

The additional risk fields now "span" the risk name field.

Aggregating Data

Aggregations are used to group and summarize numerical data.

The techniques described here are used not only to produce the desired results but to also cause efficient SQL to be generated by CommandCenter.

Follow this procedure to count the number of "Effective", "Ineffective" and "Not Determined" controls that are directly present under an entity, and then total the percentage of "Ineffective" controls for that entity. Do the following:

Procedure

1. Create a new list report.
2. Create a child query:
   a. From the **Query Explorer**, open the query.
   b. In the Properties pane under **Miscellaneous**, set the name to **listMain**.
   c. Return to the **Query Explorer** and select the top level folder **Queries**.
   d. Click the **Toolbox** tab.
   e. Drag a Query **Query** object next to the **listMain** query.
   f. Select the new query object.
   g. In the Properties pane under **Miscellaneous**, set the **Name** to **listMain1**.
3. Add another child query object to the query object you created in the previous step.
   a. Drag a Query **Query** object next to the **listMain1** query.
   b. Select the new query object.
   c. In the Properties pane under **Miscellaneous**, set the **Name** to **listMain2**.
4. Now you will add query items to the **listMain2** query you created in the previous step.
   a. Select the **listMain2** query to open it.
   b. Click the **Source** tab in the Insertable Objects pane.
   c. Drag the following query items into the **Data Items** box:
      - `DEFAULT|[DEFAULT_REL]|GRC_OBJECTS|SOXBUSENTITY_FOLDER| [SOXBUSENTITY_GPC]|[CEN_NAME00]`
      - `DEFAULT|[DEFAULT_REL]|GRC_OBJECTS|SOXBUSENTITY_FOLDER| [SOXBUSENTITY_GPC]|ID_FIELDS|[CEN_ENTITY_ID]`
      - `DEFAULT|[DEFAULT_REL]|GRC_OBJECTS|[SOXCONTROL]|ID_FIELDS| [CN_CONTROL_ID]`
      - `DEFAULT|[DEFAULT_REL]|GRC_OBJECTS|[SOXCONTROL]|ENUMERATION_FIELDS| OPERATING_EFFECTIVENESS (ENUMERATION)| [CN_OPERATING_EFFECTIVENES_ID]`
5. Now add a data item to the query object:
   a. Click the **Toolbox** tab.
   b. Drag a Data Item **Data Item** object into the **Data Items** box of the query.
   c. In the Expression Definition pane of the data item, enter the following expression:

```
if([CN_OPERATING_EFFECTIVENES_ID] + 0 = #$SOXCONTROL_OPERATING_EFFECTIVENES
_DEFINITION_MAP{"Effective"}#)
  then(1)
  else (0)
```

   d. ✓ Validate the expression and then click **OK**.
6. In the Properties pane of the data item:
   a. Change the **Name** property of the data item to **Effective**.
   b. Change the **Aggregate Function** property to **None**.
7. Repeat Steps 5 and 6 to create two more data items: one for **Ineffective** and another for **Not Determined**. Make sure you change the parameter map's **key** value accordingly for each new data item.
8. From the **Run** menu, click **View Tabular Data** to view the **listMain2** data.
9. The report is now executed only at the level of the **listMain2** object. Note that the columns **Effective**, **Ineffective**, and **Undetermined** are populated by 1s and 0s depending on the control's operating effectiveness.

10. Now you will aggregate the data in the **listMain1** object by counting the total number of controls based on their operating effectiveness for each entity:
    a. From the **Query Explorer**, click the **listMain1** query.
    b. Click the **Source** tab and drag and drop the `[CEN_ENTITY_ID]` and `[CEN_NAME00]` query items to the **Data Items** box of **listMain2**.
    c. Change the **Aggregate Function** property for each data item to **None**.
    d. Click the **Toolbox** tab and then drag a Data Item **Data Item** object into the **Data Items** box of the tabular model.
    e. In the Expression Definition pane, enter the following expression:

    ```
    total([listMain2].[Effective] for [listMain2].[CEN_ENTITY_ID])
    ```
11. In the Properties pane of the newly created data item:
    a. Change the **Name** property of the data item to **Effective Count**.
    b. Change the **Aggregate Function** property to **None**.
12. Repeat Steps 9 and 10 to create two more data items for counting controls: one for **Ineffective** and another for **Not Determined**.
13. To calculate the percentage of Ineffective controls:
    a. Click the **Toolbox** tab, and then drag a Data Item **Data Item** into the **Data Items** box.
14. In the Properties pane of the newly created data item:
    a. Change the **Name** property to **Percent Ineffective**.
    b. Change the **Aggregate Function** property to **None**.
    c. Change the **Rollup Aggregate Function** property to **None**.
    d. In the Expression Definition pane, create the following expression:

    ```
    [Ineffective Count] / ([Effective Count] +
    [Ineffective Count] + [Not Determined Count]))
    ```
15. From the **Run** menu, click **View Tabular Data** to view the tabular level data of the **listMain1** object.
16. Populate the **listMain** query as follows:
    a. From **Query Explorer** select the **listMain** query.
    b. From the **Insertable Objects** **Source** tab, drag the following query items into the **Data Items** pane of the query:
       - `[CEN_NAME00]`
       - `[CEN_ENTITY_ID]`
       - `[Effective Count]`
       - `[Ineffective Count]`
       - `[Not Determined Count]`
       - `[Percent Ineffective]`
17. In the Properties pane of each of the above data items:
    a. Change the **Aggregate Function** property to **None**.
18. Now you will populate the report page with query items:
    a. From the **Page Explorer**, click the report page to open it.
    b. From the Insertable Objects pane, select the **Query Items** tab, and then drag and drop the following query items into the list object from **listMain**:
       - [CEN_NAME00]
       - [CEN_ENTITY_ID]
       - [Effective Count]

- [Ineffective Count]
- [Undetermined Count]
- [Percent Ineffective]

c. Select the [Percent Ineffective] column in the list object.

d. Double-click the **Data Format** box in the Properties pane.

e. Set the **Format Type** to **Percent**.

f. Set the **Percentage Symbol** property to **%** (percent symbol).

g. Click **Ok**.

19. Run the report.

Results

The report appears to have duplicate rows for various business entities. Based upon the [CEN_ENTITY_ID] field, they are actually entities attached to different leaf nodes of the business entity hierarchy.

A better way to display the name of the row would be to use the full_path variable instead of the child name. The full_path variable displays the entire path from the business entity to the leaf node with all of the nodes in between.

Using Conditional Formatting

The following sections describe how to use conditional formatting.

Using Conditional Formatting to Indent Entries

Follow this procedure to indent entity names depending on their level.

Procedure

1. Create a new list report.
2. From the **Query Explorer**, open the query.
   a. In the Properties pane under **Miscellaneous**, set the name to **listMain**.
3. Add the following query items:
   - DEFAULT|[DEFAULT_REL]|GRC_OBJECTS|SOXBUSENTITY_FOLDER|[SOXBUSENTITY_GPC]|[CEN_NAME00]
   - DEFAULT|[DEFAULT_REL]|GRC_OBJECTS|SOXBUSENTITY_FOLDER|[SOXBUSENTITY_GPC]|ID_FIELDS|[CEN_ENTITY_ID]
   - DEFAULT|[DEFAULT_REL]|GRC_OBJECTS|SOXBUSENTITY_FOLDER|[SOXBUSENTITY_GPC]|[CEN_FULL_PATH]
4. Cut the following query items from the report:
   - [CEN_ENTITY_ID]
   - [CEN_FULL_PATH]
5. Set the sort order on the [CEN_FULL_PATH] data item.
   a. Select any item in the list object.
   b. Click the sorting icon on the toolbar, and then click **Advanced Sorting**.
   c. In the Grouping & Sorting dialog, drag and drop [CEN_FULL_PATH] under the **Detail Sort List** folder.
   d. Click **Ok**.
6. Select the **listMain** query from the **Query Explorer**.
7. Click the **Toolbox** tab and drag a **Data Item** into the **Data Items** pane.

Note: If you have an IBM DB2 database, you must replace the Oracle INSTR function in the following sample expression with the corresponding DB2 function.

In the **Expression Definition** pane, enter the following expression:

```
case INSTR([CEN_FULL_PATH], '/', 1, 2)
   when 0 then 1
   else
   (case INSTR([CEN_FULL_PATH], '/', 1, 3)
      when 0 then 2
      else
      (case INSTR([CEN_FULL_PATH], '/', 1, 4)
         when 0 then 3
         else
         (case INSTR([CEN_FULL_PATH], '/', 1, 5)
            when 0 then 4
            else
            (case INSTR([CEN_FULL_PATH], '/', 1, 6)
               when 0 then 5
               else
               (case INSTR([CEN_FULL_PATH], '/', 1, 7)
                  when 0 then 6
                  else
                  (case INSTR([CEN_FULL_PATH], '/', 1, 8)
                     when 0 then 7
                  end)
               end)
            end)
         end)
      end)
   end)
End
```

8. In the **Properties** pane, set the **Name** property of the new data item to Level.
9. Now you need to make the **Level** data item a property of the list.
   a. Select the **List** object from the main report page.
   b. In the Properties pane under **Data** double-click on **Properties**.
   c. Select [Level].
   d. Click **OK**
10. Open the **Condition Explorer** and click on **Variables**.
11. In the Variables dialog:
   a. Drag and drop a String Variable **String Variable** object to the Variables pane.
   b. Drag the query item **Level** into the Expression Definition pane.
   c. Click **OK**.
12. In the Properties pane under **Miscellaneous**, set the **Name** property to **Indent Variable**.
13. In the **Values pane**, do the following:
   a. Click the **Add** icon and enter **1** in the dialog box.
   b. Repeat step a for all the levels 2-7.
14. Select the main report page from the **Page Explorer**.
15. In the report page, select [CEN_NAME00] and in the **Properties** box do the following:
   a. Click the **Style Variable** property and select the variable **Indent Variable**.
   b. Click **OK**.
16. Increase the left indentation for different levels of entities:
   a. Select the [CEN_NAME00] column.

   b. Hover the mouse pointer over the **Conditional Explorer** bar.
   c. Select the variable value **2** that is present under variable **Indent Variable**. The explorer bar will turn green indicating that you are setting the value of a variable.
   d. In the Properties pane under **Box**, click the **Padding** property and enter value of 50 px for the left padding cell, and the click **OK**.
   e. Repeat Steps c-d and select different values of the **Indent Variable** and enter increasing values for the left padding.
   f. In the Condition Explorer click **(No variable)**.

17. Run the report.

Highlighting Data Conditionally

You can conditionally highlight some rows of data when that data meets certain criteria.

The following example is based upon the report that was created in the “Aggregating Data” on page 75. This example shows you how to conditionally highlight the `[Percent Ineffective]` column.

In this example, the background color of a cell changes based upon the percentage of an ineffective control. If the percentage of the controls that are ineffective is greater than 50, the background color of the cell will be red; if the percentage is between 20 and 50 then the background color of the cell will be yellow, if it is less than 20, then the background color of the cell will be green.

Procedure

1. Open the report you created in the “Aggregating Data” on page 75.
2. Open the **Condition Explorer** and click on **Variables**.
3. In the Variables dialog:
   a. Drag and drop a **String Variable** object to the Variables pane.
   b. In the Expression Definition pane enter the following expression:
   ```
   if([Percent Ineffective] >= 0.5)
   then (3)
   else if ( [Percent Ineffective] < 0.5 and [Percent Ineffective] > 0.2)
   then (2)
   else if ( [Percent Ineffective] <= 0.2)
   then (1)
   else (0)
   ```
   c. Click **OK**.
4. In the Properties pane under **Miscellaneous**, set the **Name** property to **Highlight Variable**.
5. In the **Values pane**, Click the **Add** icon and enter **3** in the dialog box. Repeat for **2**, **1**, and **0**.
6. Select the main report page from the **Page Explorer**.
7. In the report page, select `[Percent Ineffective]` and in the **Properties** box do the following:
   a. Click the **Style Variable** property and select the variable **Highlight Variable**.
   b. Click **OK**.
8. Now you will set the colors for the different values:
   a. Select the `[Percent Ineffective]` column.

b. Hover the mouse pointer over the **Conditional Explorer** bar.

c. Select the variable value **1** that is present under variable **Indent Variable**. The explorer bar will turn green indicating that you are setting the value of a variable.

d. In the Properties pane under **Color & Background**, set the **Background Color** property to **Green**.

e. Repeat steps c-d for each variable value **2** and **3**, and then set the **Background color** property to **Yellow** and **Red**, respectively.

f. In the Condition Explorer, click **(No variable)**.

9. Run the report.

Results

In the report, values that are less than 20% are highlighted in green and values that are greater than 20% and less than 50% are highlighted in yellow.

Working With Bar Chart Colors

When representing data in bar charts, report authors should try to control colors for consistency so that particular colors mean the same thing on a given report each time users run that report.

This section contrasts two techniques you can use for creating bar chart reports. To illustrate the results of bar chart colors from each method, the control object's "Operating Effectiveness" field is used with all or some of its corresponding values (Effective, Ineffective, Not Determined) in the examples that follow.

- Method 1: This method automatically displays values that are added to the 'Operating Effectiveness' field in the schema (values are not mapped). However, the colors in the reports for these values are inconsistently displayed.

  For instance, there are two business entities: BE1 and BE2. The "Operating Effectiveness" field for these business entities are as follows:

  - BE1 - Effective, Ineffective, Not Determined
  - BE2 - Ineffective, Not Determined

    When a user runs the report for each business entity, the bar colors that result for the values are inconsistent:

  - BE1 - Effective = black, Ineffective = pink, Not Determined = purple
  - BE2 - Ineffective = black, Not Determined = pink

    For details, see "Displaying Bar Color Variation with Automatically Displayed Values" on page 82.

- Method 2: This method uses a parameter mapping technique and requires the bar chart report to be modified when new values are added, for example, to the "Operating Effectiveness" field in the schema. However, colors are consistently displayed in the reports for the values that are mapped.

  Using the same two business entities with the same values in the "Operating Effectiveness" as in the example in Method 1, the results using Method 2 are different.

  When a user runs the report for each business entity, the bar colors that result for the values are consistent:

  - BE1 - Effective = black, Ineffective = pink, Not Determined = purple
  - BE2 - Ineffective = pink, Not Determined = purple

For details, see "Displaying Bar Color Consistency with Report Modification for New Values."

Displaying Bar Color Variation with Automatically Displayed Values

Procedure

1. Create a new Chart report.
   a. Select the **Column** Chart grouping.
   b. Select the **Column** chart type.
   c. Click **OK**.
2. Drag and drop the following query items into the various chart sections:
   - `Categories (x-axis)`

     ```
     DEFAULT|[DEFAULT_REL]|GRC_OBJECTS|SOXBUSENTITY_FOLDER|
     [SOXBUSENTITY_GPC]|[CEN_NAME00]
     ```
   - `Measures (y-axis)`

     ```
     DEFAULT|[DEFAULT_REL]|GRC_OBJECTS|[SOXCONTROL]|ID_FIELDS|
     [CN_CONTROL_ID]
     ```
   - `Series`

     ```
     DEFAULT|[DEFAULT_REL]|GRC_OBJECTS|[SOXCONTROL]|ENUMERATION_FIELDS|
     OPERATING_EFFECTIVENES (ENUMERATION)|[CN_OPERATING_EFFECTIVENES]
     ```
3. From the **Query Explorer**, select the query.
   a. In the Properties pane under **Miscellaneous**, set the name to **chartMain**.
4. In the **Data Items** pane, select `[CN_CONTROL_ID]`**.**
   a. In the Properties pane under **Data Item**, change the value of the **Aggregate Function** property to **Count Distinct**.
   b. In the Properties pane under **Data Item**, change the value of the **Rollup Aggregate Function** property to **Automatic**.
5. From the **Page Explorer**, create a **Prompt Page** and create a **Business Entity** prompt as shown in "Adding a Business Entity Prompt" on page 56.
6. Double-click on the title and set the value to **Operating Effectiveness**.
7. Open the **Axis Titles** and do the following:
   a. Select the x-axis title and in the Properties pane.
   b. Under **General** set the **Default Title** property to **No**.
   c. Double-click on the axis title and set the value to **Business Entity**.
   d. Repeat steps a-c for the y-axis and set the value to **Control Count**.
8. Select the **(Default Legend Title)** and do the following:
   a. In the Properties pane under **General** set the **Default Title** property to **No**.
   b. Double-click on the legend title and set the value to **Control Operating Effectiveness**.
9. Run the report.

Displaying Bar Color Consistency with Report Modification for New Values

Procedure

1. Create a new chart report.
   a. Select the **Column** Chart grouping.
   b. Select the **Column** chart type.

2. Create a child query:
   a. From the **Query Explorer**, select queries.
   b. Select **Query1** and in the Properties pane under **Miscellaneous**, set the name to **listMain**.
   c. Click the **Toolbox** tab.
   d. Drag a **Query** object onto the **graphMain** query.
   e. Select the new query object.
   f. In the Properties pane under **Miscellaneous**, set the **Name** to **graphMain1**.
3. Now you will add query items to the **graphMain1** query you created in the previous step.
   a. Select the **graphMain1** query to open it.
   b. Click the **Source** tab in the Insertable Objects pane.
   c. Drag the following query items into the **Data Items** box:
      - `DEFAULT|[DEFAULT_REL]|GRC_OBJECTS|SOXBUSENTITY_FOLDER| [SOXBUSENTITY_GPC]|[CEN_NAME00]`
      - `DEFAULT|[DEFAULT_REL]|GRC_OBJECTS|SOXBUSENTITY_FOLDER| [SOXBUSENTITY_GPC]|ID_FIELDS| [CEN_ENTITY_ID]`
      - `DEFAULT|[DEFAULT_REL]|GRC_OBJECTS|[SOXCONTROL]|ID_FIELDS| [CN_CONTROL_ID]`
      - `DEFAULT|[DEFAULT_REL]|GRC_OBJECTS|[SOXCONTROL]|ENUMERATION_FIELDS| OPERATING_EFFECTIVENES (ENUMERATION)| [CN_OPERATING_EFFECTIVENES_ID]`
4. Now do the following:
   a. Click the **Toolbox** tab.
   b. Drag a **Data Item** object into the **Data Items** box of the query.
   c. In the Expression Definition pane of the data item, enter the following expression:

```
if([CN_OPERATING_EFFECTIVENES_ID] = #$SOXCONTROL_OPERATING_EFFECTIVENES
_DEFINITION_MAP{"Effective"}#)
   then(1)
   else (0)
```

   d. Validate the expression and then click **OK**.
5. In the Properties pane of the data item:
   a. Change the **Name** property of the data item to **Effective**.
   b. Change the **Aggregate Function** property to **None**.
6. Repeat Steps 5 and 6 to create two more data items: one for **Ineffective** and another for **Not Determined**. Make sure you change the parameter map's **key** value accordingly for each new data item.
7. From the **Run** menu, click **View Tabular Data** to view the **graphMain1** data.
8. The report is now executed only at the level of the **graphMain1** object. Note that the columns **Effective**, **Ineffective** and **Undetermined** are populated by 1s and 0s depending on the control's operating effectiveness.
9. Now you will aggregate the data in the **graphMain1** object by counting the total number of controls based on their operating effectiveness for each entity:
   a. From the **Query Explorer**, click the **graphMain** query.
   b. Drag and drop the `[CEN_ENTITY_ID]` and `[CEN_NAME00]` query items to the **Data Items** box of **graphMain**.
   c. Change the **Aggregate Function** property for each data item to **None**.
   d. Drag and drop the `[Effective]`, `[Ineffective]` and [Not Determined] query items to the **Data Items** box of **graphMain**.
   e. Change the **Aggregate Function** property for each data item to **Total**.

10. From the **Page Explorer**, create a **Prompt Page** and create a **Business Entity** prompt as shown in "Adding a Business Entity Prompt" on page 56.
11. Drag and drop the following query items into the various chart sections:
    - `Categories (x-axis)`

      ```
      [CEN_NAME00]
      ```
    - `Measures (y-axis)`

      ```
      [Effective]
      [Ineffective]
      [Not Determined]
      ```
12. Double-click on the title and set the value to **Operating Effectiveness**.
13. Open the **Axis Titles** and do the following:
    a. Select the x-axis title.
    b. In the Properties pane, under **General** set the **Default Title** property to **No**.
    c. Double-click on the axis title and set the value to **Business Entity**.
    d. Repeat steps a-c for the y-axis and set the value to **Control Count**.
14. Select the **(Default Legend Title)** and do the following:
    a. In the Properties pane under **General** set the **Default Title** property to **No**.
    b. Double-click on the legend title and set the value to **Control Operating Effectiveness**.
15. Run the report.

Editing Bar Colors

Follow this procedure to set the colors that the chart will use.

Procedure

1. Select the **Combination Graph** object in the report.
2. In the Properties pane under the **Color & Background** category, click on the **Palette** property.
3. Using the ⬆ **Up** and ⬇ **Down** arrows, move the **Green**, **Yellow**, and **Red** bars to the top of the palette.
4. Click **OK**.
5. Run the Report.

Creating Joins

The section will describe how to create the two types of joins that are common when creating reports: direct joins where we only obtain secondary objects that are directly related to the primary object and indirect joins where we see all the secondary objects that can roll-up to the primary object.

In the IBM OpenPages GRC on Cloud V6 framework there are no longer query subjects that join primary objects to secondary objects as was in the legacy framework. This was done to save time when generating the framework and due to the fact that we could never generate all the possible query subjects before reaching the maximum allowable framework size in IBM Cognos.

Creating Direct Joins

Follow this procedure to create a direct join.

Start by creating a report of entity to issues.

Procedure

1. Create a new list report.
2. In the **Query Explorer**, select queries.
3. Drag and drop two additional Query Queries into the right pane.
4. Name the queries as follows: listMain, Entity and Issue.
5. Select the Issue query.
6. Drag the following query items into the **Data Items** box:
   - `DEFAULT|[DEFAULT_REL]|GRC_OBJECTS_STANDALONE|[SOXISSUE]|[IS_NAME00]`
   - `DEFAULT|[DEFAULT_REL]|GRC_OBJECTS_STANDALONE|[SOXISSUE]|[IS_DESCRIPTION]`
   - `DEFAULT|[DEFAULT_REL]|GRC_OBJECTS_STANDALONE|[SOXISSUE]|PARENT_CONTEXT|SOXBUSENTITY_CHILD|[CEN_ENTITY_ID]`
   - `DEFAULT|[DEFAULT_REL]|GRC_OBJECTS_STANDALONE|[SOXISSUE]|PARENT_CONTEXT|SOXBUSENTITY_CHILD|[CEN_REPORTING_PERIOD_ID]`
7. Select the Entity query in the **Query Explorer**.
8. Drag the following query items into the **Data Items** box:
   - `DEFAULT|[DEFAULT_REL]|GRC_OBJECTS|SOXBUSENTITY_FOLDER|[SOXBUSENTITY_GPC]|[GEN_NAME00]`
   - `DEFAULT|[DEFAULT_REL]|GRC_OBJECTS|SOXBUSENTITY_FOLDER|[SOXBUSENTITY_GPC]|[GEN_FULL_PATH]`
   - `DEFAULT|[DEFAULT_REL]|GRC_OBJECTS|SOXBUSENTITY_FOLDER|[SOXBUSENTITY_GPC]|ID_FIELDS|[CEN_ENTITY_ID]`
   - `DEFAULT|[DEFAULT_REL]|GRC_OBJECTS|SOXBUSENTITY_FOLDER|[SOXBUSENTITY_GPC]|ID_FIELDS|[CEN_REPORTING_PERIOD_ID]`
9. In the **Query Explorer**, select queries.
10. Drag and drop a Join join next to the listMain query and then populate the join with the Entity and Issue queries.
11. Double click on the join and do the following:
    a. Click **New Link**.
    b. Select `[CEN_REPORTING_PERIOD_ID]` in both columns.
    c. Click **New Link.**
    d. Select `[CEN_ENTITY_ID]` in both columns.
    e. Click **OK**.
12. Select the listMain query.
13. Drag and drop the following query items from the Insertable Objects pane:
    a. GEN_FULL_PATH
    b. IS_NAME00
    c. IS_DESCRIPTION
14. Set the properties **Aggregate Function** and **Rollup Aggregate Function** to None for all three data items.
15. Go to the main report page and place the 3 data items in the list.
16. Run the report.

Results

What you see in the report is the following:

- Only issues that are directly joined to entities are shown.
- There are only actually two issues here, ISS-012-01 joined to the entity Agency Services under EMEA and ISS-012-03 joined to the entity Agency Services under Asia Pac.
- Since we are showing the grandparent full path we see all entities that roll-up to the top level entity of Global financial Services. This makes it appear that issues are attached at three locations but they are not.
- If we display the child full path instead of the grandparent full path we only see the two issues directly attached to their entities.

Creating Indirect Joins

Indirect joins are used to obtain all secondary objects that are both directly related to the primary object type and indirectly related to the primary object type.

In “Creating Direct Joins” on page 85 we could only obtain issues directly attached to entities. In this task we will obtain all issues directly attached to entities and any other object that has issues that are attached to entities.

Procedure

1. Create a new list report.
2. In the **Query Explorer**, select queries.
3. Drag and drop to additional Query Queries into the right pane.
4. Name the queries as follows: listMain, Entity and Issue.
5. Select the Issue query.
6. Drag the following query items into the **Data Items** box:
   - `DEFAULT|[DEFAULT_REL]|GRC_OBJECTS_STANDALONE|[SOXISSUE]|[IS_NAME00]`
   - `DEFAULT|[DEFAULT_REL]|GRC_OBJECTS_STANDALONE|[SOXISSUE]|`
   - `DEFAULT|[DEFAULT_REL]|GRC_OBJECTS_STANDALONE|[SOXISSUE]|ID_FIELDS|[IS_ISSUE_ID]`
   - `DEFAULT|[DEFAULT_REL]|GRC_OBJECTS_STANDALONE|[SOXISSUE]|ID_FIELDS|[IS_REPORTING_PERIOD_ID]`
7. Select the Entity query in the **Query Explorer**.
8. Drag the following query items into the **Data Items** box:
   - `DEFAULT|[DEFAULT_REL]|GRC_OBJECTS|SOXBUSENTITY_FOLDER|[SOXBUSENTITY_GPC]|[GEN_NAME00]`
   - `DEFAULT|[DEFAULT_REL]|GRC_OBJECTS|SOXBUSENTITY_FOLDER|[SOXBUSENTITY_GPC]|[GEN_FULL_PATH]`
   - `DEFAULT|[DEFAULT_REL]|GRC_OBJECTS|SOXBUSENTITY_FOLDER|[SOXBUSENTITY_GPC]|ID_FIELDS|[CEN_ENTITY_ID]`
9. In the **Query Explorer**, select queries.
10. Drag and drop a Join join next to the listMain query and then populate the join with the Entity and Issue queries.
11. Double click on the join and do the following:
    a. Click **Convert to Expression**.
    b. Enter the following in the Expression Definition pane.

```
"OP_RPS_AUX.IS_REL_PARENT_CHILD"(
[Entity].[CEN_ENTITY_ID],
[Issue].[IS_ISSUE_ID],
[Entity].[CEN_REPORTING_PERIOD_ID]) = 'Y' and
[Entity].[CEN_REPORTING_PERIOD_ID] = [Issue].[IS_REPORTING_PERIOD_ID]
```

   c. Click **OK**.

12. Select the listMain query.
13. Drag and drop the following query items from the Insertable Objects pane:
    a. GEN_FULL_PATH
    b. IS_NAME00
    c. IS_DESCRIPTION
14. Set the properties **Aggregate Function** and **Rollup Aggregate Function** to None for all three data items.
15. Goto the main report page and place the 3 data items in the list.
16. Run the report.

Results

There are many more issues attached to the top level Global Financial Services entity because any issue that can be related through any object attached to an entity will be shown because of how we changed the join.

Tip: If you create an indirect join, the data returned will be from the object model defined in the application and not be limited by the objects in the namespace.

Creating Drill-Through Reports

Follow this procedure to create drill-through report.

You can attach different sub-reports to many different places in a report such as the legend or a bar of a graph. You can also attach multiple drill-throughs to the same data item and give the user a choice of which one to select.

Important: Relational reports can only drill-through to another relational report.

Procedure

1. Create a new list report.
2. From the Insertable Objects pane, drag and drop the following data items into the list (please remember that what is loaded in your system will be different then what you see here):
   - DEFAULT|[DEFAULT_REL]|GRC_OBJECTS|SOXBUSENTITY_FOLDER| [SOXBUSENTITY_GPC]|[PEN_NAME00]
   - DEFAULT|[DEFAULT_REL]|GRC_OBJECTS|[SOXCONTROL]|[CN_NAME00]
   - DEFAULT|[DEFAULT_REL]|GRC_OBJECTS|[SOXCONTROL]|[CN_DESCRIPTION]
   - DEFAULT|[DEFAULT_REL][DEFAULT_REL]|GRC_OBJECTS|[SOXCONTROL] ENUMERATION_FIELDS|OPERATING_EFFECTIVENES(ENUMERATION)| [CN_OPERATING_EFFECTIVENES]
   - DEFAULT|[DEFAULT_REL]|GRC_OBJECTS|[SOXCONTROL] ENUMERATION_FIELDS| OPERATING_EFFECTIVENES(ENUMERATION)| [CN_OPERATING_EFFECTIVENES_ID]
3. Cut the CN_OPERATING_EFFECTIVENES_ID field from the list.
4. Go to the **Query Explorer** and add the following filters to the query:
   - [DEFAULT_REL].[SOXBUSENTITY_GPC].[GEN_ENTITY_ID] = ?Entity ID?
   - [CN_OPERATING_EFFECTIVENES_ID] = ?Operating Effectiveness ID?

- `[DEFAULT_REL].[SOXBUSENTITY_GPC].[GEN_REPORTING_PERIOD_ID] = -1`
5. Validate the report.
6. Save the report with the name **Drill Through Child**.

What to do next

"Creating the Parent Report"

Creating the Parent Report

Follow this procedure to create a parent report.

Procedure

1. Create a new crosstab report.
2. Add a business entity prompt as described "Adding a Business Entity Prompt" on page 56. Use GEN_ENTITY_ID as the field for the source of the data. Your crosstab query should have the following filter after creating the prompt:
   `[DEFAULT_REL].[SOXBUSENTITY_GPC].[GEN_ENTITY_ID] = ?Entity ID?`
3. Using the **Query Explorer**, select the query for the crosstab.
4. From the Insertable Objects pane, drag and drop the following data items into the Data Items pane of the crosstab:
   - `DEFAULT|[DEFAULT_REL]|GRC_OBJECTS|SOXBUSENTITY_FOLDER| [SOXBUSENTITY_GPC]| [PEN_NAME00]`
   - `DEFAULT|[DEFAULT_REL]|GRC_OBJECTS|SOXBUSENTITY_FOLDER| [SOXBUSENTITY_GPC]| ID_FIELDS|[PEN_ENTITY_ID]`
   - `DEFAULT|[DEFAULT_REL]|GRC_OBJECTS|[SOXCONTROL]|ID_FIELDS| [CN_CONTROL_ID]`
   - `DEFAULT|[DEFAULT_REL]|GRC_OBJECTS|[SOXCONTROL]|ENUMERATION_FIELDS| OPERATING_EFFECTIVENES (ENUMERATION)|[CN_OPERATING_EFFECTIVENES]`
   - `DEFAULT|[DEFAULT_REL]|GRC_OBJECTS|[SOXCONTROL]|ENUMERATION_FIELDS| OPERATING_EFFECTIVENES (ENUMERATION)| [CN_OPERATING_EFFECTIVENES_ID]`
5. Select `CN_CONTROL_ID and in the Properties Pane and do the following:`
   a. Under the **Data Item** category, set the **Aggregate Function** property to **Count Distinct**.
   b. Under the **Data Item** category, set the **Rollup Aggregate Function** to **Automatic**.
6. Return to the main page of the report.
7. Now you will do the following to populate the crosstab:
   a. Drag and drop the [CN_OPERATING_EFFECTIVENES] field into the columns area from the **Data Items** tab of the Insertable Objects pane.
   b. Drag and drop the [PEN_NAME00] field into the rows area from the **Data Items** tab of the Insertable Objects pane.
   c. Drag and drop the [CN_CONTROL_ID] field into the measures area from the **Data Items** tab
8. Perform the following formatting:
   a. Center the column titles.
   b. Center the crosstab intersection.
   c. Set the sort order on the columns to ascending.
   d. Set the sort order on the rows to ascending.
9. Run the report.

What to do next

"Adding a Drill-Through Report to the Parent"

Adding a Drill-Through Report to the Parent

Follow this procedure to add a drill-through report to a report so the user can get more detailed information on the controls behind the numbers in the crosstab.

Procedure

1. Select the main page of a report.
2. Unlock the crosstab.
3. Select the column header CN_OPERATING_EFFECTIVENES and do the following:
   a. In the Properties pane, under the **Data** category, select **Properties**.
   b. Check CN_OPERATING_EFFECTIVENES_ID so it can be used as a pass through parameter to the sub-report.
   c. Click **OK**.
4. Select the row header PEN_NAME00 and do the following:
   a. In the Properties pane, under the **Data** category, select **Properties**.
   b. Check PEN_ENTITY_ID so it can be used as a pass through parameter to the sub-report.
   c. Click **OK.**
5. Select the measure in the crosstab.
6. In the Properties pane, under **Data**, select **Drill-Through Definitions**.
7. Add a new drill-through definition.
8. Set the name of the drill-through to **Control Detail**.
9. Select the report **Drill Through Child**.
10. Check to open the report in a new window.
11. Click ✎ to edit the parameters passed to the sub-report. Do the following:
    a. For **Entity ID**, set the method to **Pass Data Item Value** and the **Value** to **PEN_ENTITY_ID**
    b. For **Operating Effectiveness ID**, set the method to **Pass Data Item Value** and the **Value** to **CN_OPERATING_EFFECTIVENES_ID**.
    c. Click **OK**.
12. Run the report.

Results

The measures in the crosstabs now appear blue and underlined indicating a drill-through is attached to that cell. When you click on the intersection of Asia Pac and Not Determined, a drill-through report appears.

The drill-through report shows three records with the operating effectiveness of "Not Determined" that came from the Asia Pac business entity.

Creating a Master-Detail Report

A master-detail report typically contains a second list embedded in the main report list. This creates a situation where you have an outer query for the main report list and a sub-query for the embedded list. What this means in terms of performance is that for each record pulled back from the database another query will be run. You should only use this construct if it is your only choice because of the structure of the data.

Procedure

1. Create a new list report.
2. Add a business entity prompt as described "Adding a Business Entity Prompt" on page 56. Use GEN_ENTITY_ID as the field for the source of the data. Your crosstab query should have the following filter after creating the prompt:
   ```
   [DEFAULT_REL].[SOXBUSENTITY_GPC].[GEN_ENTITY_ID] = ?Entity ID?
   ```
3. From the Insertable Objects pane, drag and drop the following data items into the list (please remember that what is loaded in your system may be different then what you see here):
   - `DEFAULT|[DEFAULT_REL]|GRC_OBJECTS|[SOXCONTROL]|ID_FIELDS| [CN_CONTROL_ID]`
   - `DEFAULT|[DEFAULT_REL]|GRC_OBJECTS|[SOXCONTROL]|[CN_NAME00]`
   - `DEFAULT|[DEFAULT_REL]|GRC_OBJECTS|[SOXCONTROL]|[CN_DESCRIPTION]`
   - `DEFAULT|[DEFAULT_REL]|GRC_OBJECTS|[SOXCONTROL] ENUMERATION_FIELDS| OPERATING_EFFECTIVENES(ENUMERATION)| [CN_OPERATING_EFFECTIVENES]`
4. Cut the CN_CONTROL_ID field from the list.
5. Go to the **Query Explorer** and add the following filters to the query:
   ```
   [DEFAULT_REL].[SOXBUSENTITY_GPC].[GEN_REPORTING_PERIOD_ID] = -1
   ```
6. Validate the report.
7. Go to the main report page and do the following:
   a. Sort CN_NAME00 ascending.
   b. From the Insertable Objects pane Toolbox tab, select a list and drop it into the last column of the existing list. Remember the list must be locked to do this.
8. In the new list drag and drop the following data items:
   - `DEFAULT|[DEFAULT_REL]|GRC_OBJECTS|[SOXCONTROL]|ID_FIELDS| [CN_CONTROL_ID]`
   - `DEFAULT|[DEFAULT_REL]|GRC_OBJECTS|[SOXCONTROL]|ENUMERATION_FIELDS| FINANCIAL_STATEMENT_AS(ENUMERATION)|[CN_FINANCIAL_STATEMENT_AS]`
9. Run the report.

What to do next

"Creating Relationships Between Master and Embedded Lists"

Creating Relationships Between Master and Embedded Lists

The report is obviously incorrect because the CN_CONTROL_ID values do not match. This is because we have not created the relationship between the master list and the embedded list.

Follow this procedure to create the relationship.

Procedure

1. Select an item in the embedded list.
2. From the **Data** menu on the main toolbar, select **Master Detail Relationships**.
3. Create a join between CN_CONTROL_ID in the listMain and the CN_CONTROL_ID in the childList.
4. Click **OK**.
5. Run the report.

Results

Now there are fewer rows in the embedded list and the control ID values match the master record.

What to do next

"Formatting Embedded Lists"

Formatting Embedded Lists

Now let's do some cleanup activities to make the embedded list less obvious. Do the following:

Procedure

1. Select the CN_CONTROL_ID fields in both lists and cut them. They need to be in the query but don't need to be displayed.
2. Select the header of the embedded list and do the following:
   a. Using the ancestor selector, select list.
   b. In the **Properties** pane under **General**, set the **Column Titles** property to **Hide**.
3. Select the cell in the main list that the embedded list sits within. In the **Properties** pane under **Box**, select the **Padding** property and set all the values to **zero**.
4. Run the report.

Adding Microcharts

A microchart is simply a miniature version of a chart that can be inserted in a crosstab or a list. They operate in the same manner that a master-detail report functions in that the microchart is a detailed view of the master data.

Follow this procedure to create a report with a microchart.

Procedure

1. Create a new list report.
2. Add a business entity prompt as described "Adding a Business Entity Prompt" on page 56. Use GEN_ENTITY_ID as the field for the source of the data. Your crosstab query should have the following filter after creating the prompt:
   ```
   [DEFAULT_REL].[SOXBUSENTITY_GPC].[GEN_ENTITY_ID] = ?Entity ID?
   ```
3. From the Insertable Objects pane, drag and drop the following data items into the list (please remember that what is loaded in your system will be different then what you see here):

- `DEFAULT|[DEFAULT_REL]|GRC_OBJECTS|SOXBUSENTITY_FOLDER| [SOXBUSENTITY_GPC]|[PEN_NAME00]`
- `DEFAULT|[DEFAULT_REL]|GRC_OBJECTS|SOXBUSENTITY_FOLDER| [SOXBUSENTITY_GPC]|ID_FIELDS|    [PEN_ENTITY_ID]`
- `DEFAULT|[DEFAULT_REL]|GRC_OBJECTS|[SOXCONTROL] ENUMERATION_FIELDS| OPERATING_EFFECTIVENES(ENUMERATION)| [CN_OPERATING_EFFECTIVENES]`

4. Cut the `PEN_ENTITY_ID` field and the `CN_OPERATING_EFFECTIVENES field` from the list.
5. Go to the **Query Explorer** and add the following filters to the query:
   - `[DEFAULT_REL].[SOXBUSENTITY_GPC].[GEN_REPORTING_PERIOD_ID] = -1`
   - `[CN_OPERATING_EFFECTIVENES] is not null`
6. In the **Query Explorer**, rename the query used by the list to **listMain**.
7. Validate the report.
8. Go to the main report page and do the following:
   a. Sort PEN_NAME00 ascending.
   b. Set the label for PEN_NAME00 to **Business Entity**.
   c. From the Insertable Objects pane Toolbox tab, select a Chart chart and drop it into the last column of the existing list. Remember the list must be locked to do this.
   d. Select **Microchart** and a **Column** bar chart.
9. In the new chart drag and drop the following data items:
   - `DEFAULT|[DEFAULT_REL]|GRC_OBJECTS|[SOXCONTROL]|ID_FIELDS| [CN_CONTROL_ID]`
   - `DEFAULT|[DEFAULT_REL]|GRC_OBJECTS|[SOXCONTROL] ENUMERATION_FIELDS| OPERATING_EFFECTIVENES(ENUMERATION)| [CN_OPERATING_EFFECTIVENES]`
10. In the **Query Explorer**, rename the query used by the list to **microChart**.
11. Go to the **Query Explorer** and add the following filters to the query:
    - `[DEFAULT_REL].[SOXBUSENTITY_GPC].[GEN_REPORTING_PERIOD_ID] = -1`
    - `[CN_OPERATING_EFFECTIVENES] is not null`
12. From the **Insertable Objects** pane drag and drop the following data item:
    ```
    DEFAULT|[DEFAULT_REL]|GRC_OBJECTS|SOXBUSENTITY_FOLDER|
         [SOXBUSENTITY_GPC]|ID_FIELDS|[PEN_ENTITY_ID]
    ```
13. Select the CN_CONTROL_ID data item and do the following in the Properties pane:
    a. Under the **Data Item** category, set the **Aggregate Function** property to **Count Distinct**.
    b. Under the **Data Item** category, set the **Rollup Aggregate Function** to **Automatic**.
14. On the main report page rename the chart column to **Operating Effectiveness**.
15. From the **Data** menu on the main toolbar, select **Master Detail Relationships**.
    a. Create a join between PEN_ENTITY_ID in the listMain and the PEN_ENTITY_ID in the microChart.
    b. Click **OK**.
16. Select the microchart background and do the following:
    a. In the Properties pane under the category **Positioning**, set the **Size & Overflow** property to a height of 100 px and set the width to 150 px.
17. Select the y axis of the microchart and in the Properties pane under the **General** category, set the **Axis Labels** property to **Show**.

18. In the **Condition Explorer** select variables and do the following:
    a. Add a new string variable.
    b. In the expression editor, add the following:
       `[microChart].[CN_OPERATING_EFFECTIVENES]`
    c. Click **OK**.
    d. Add three values: Effective, Ineffective and Not Determined.
    e. In the Properties pane under **Miscellaneous**, set the **Name** property to **Operating Effectiveness**
19. On the main report page, select the microchart and in the **Properties** pane under **Color & Background**, select the **Conditional Palette** and do the following:
    a. For the **Variable**, select **Operating Effectiveness**.
    b. For **Effective** set the color to **lime**.
    c. For **Ineffective** set the color to **red**.
    d. For **Not Determined** set the color to **blue**.
    e. Click **OK**.
20. Select the **Series bar**. In the Properties pane under **Chart Labels**, set the **Values** property to **Show**.
21. Run the report.

Results

There is now a comparison of control Operating Effectiveness across different business entities to compare progress in how they are getting their process under control.

Chapter 8. Working with Dimensional Data

This chapter introduces the concepts of creating dimensional reports.

The structure of the dimensional model is explained in Chapter 4, “Reporting Framework,” on page 15.

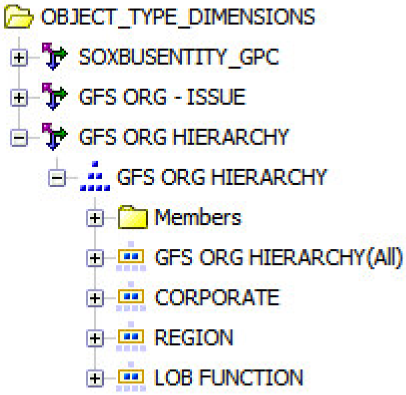
Adding Prompts

The prompt page adds interactivity for users and acts as a report scoping mechanism for filtering and limiting the retrieved data.

You can add a single or full path business entity prompt.

Adding a Single Level Business Entity Dimensional Prompt

A single level Business Entity prompt added to your report is useful for creating cascaded prompts.



```
OBJECT_TYPE_DIMENSIONS
  SOXBUSENTITY_GPC
  GFS ORG - ISSUE
  GFS ORG HIERARCHY
    GFS ORG HIERARCHY
      Members
      GFS ORG HIERARCHY(All)
      CORPORATE
      REGION
      LOB FUNCTION
```

Figure 38. Dimensional Business Entity Structure

Procedure

1. If you do not already have a prompt page in your report, follow the instructions in “Creating a Prompt Page” on page 54
2. To create your Business Entity prompt, do the following:
   a. Drag a **Text Item** into the first column of the table. Enter the text for the prompt in the text item.
   b. Select the table cell and set the location to be horizontally right and vertically top.
   c. Make the text bold using the B bold icon.
3. Drag a **Value Prompt** object into the second column of the prompt page. This will bring up the Prompt Wizard.
4. On the **Choose Parameter** screen of the Prompt Wizard:
   a. Select the option **Create a new parameter**.
   b. In the box, type a name for the parameter. For example, `Business Entity`.
   c. Click **Next**.

5. On the **Create Filter** screen:
   a. Select the option **Create a parameterized filter**.
   b. In the **Model item** box, click the browse button and navigate to the following query item (Remember your Business Entity dimension will be named differently):

      DEFAULT|[DEFAULT_DIM]|OBJECT_TYPE_DIMENSIONS| [GFS ORG HIERARCHY]| [GFS ORG HIERARCHY]| [CORPORATE]
   c. In the **Operator** box, use **=** (the equal sign is the default).
   d. Clear the box for the **Make this filter optional** option.
   e. Click **Next**.
6. If you have multiple queries in your report, in the **Apply filter** box, select the query to which the filter should be applied.
7. On the **Populate Control** screen:
   a. Select the **Create new query** option.
   b. In the **Name** box, type a name for the query. For example, **Business Entity Prompt**.
   c. In the **Values to display** box, click the browse button and navigate to the following query item:

      DEFAULT|[DEFAULT_DIM]|OBJECT_TYPE_DIMENSIONS| [GFS ORG HIERARCHY]| [GFS ORG HIERARCHY]| [CORPORATE]
   d. The following values will be displayed in the box:

      [DEFAULT_DIM].[GFS ORG HIERARCHY].[GFS ORG HIERARCHY].[CORPORATE]
8. Click **Finish**.
9. Now select the prompt you just created and under the **General** category in the Properties pane, change the **Select UI** value to **List Box**.
10. Using the **Query Explorer**, do the following:
    a. Select the filter that was just created in the main query.
    b. In the Properties pane, under **General**, set the **Application** property to **After Auto Aggregation**.
11. Run the report.

Results

There are a few differences between this Business Entity prompt and a relational prompt:

- In order to create the prompt you had to choose a level of the Business Entity object type dimension. This means the prompt will only show data at that level.
- The prompt you created is a dimensional prompt. You can only use it to filter dimensional data in your report.
- The value to use and the value to display are the same.
- The filter appears in the Detail Filters pane of the query.

Adding a Full Path Business Entity Prompt

Follow this procedure to add a full path Business Entity prompt to your report.

Procedure

1. If you do not already have a prompt page in your report, follow the instructions in “Creating a Prompt Page” on page 54.
2. To create your Business Entity prompt, do the following:
   a. Drag a **Text Item** into the first column of the table. Enter the text for the prompt in the text item.
   b. Select the table cell and set the location to be horizontally right and vertically top.
   c. Make the text bold using the **B** bold icon.
   d. Drag a **Value Prompt** object onto your prompt page. This will bring up the Prompt Wizard.
3. On the **Choose Parameter** screen of the Prompt Wizard:
   a. Select the option **Create a new parameter**.
   b. In the box, type a name for the parameter. For example, **Business Entity**.
   c. Click **Next**.
4. On the **Create Filter** screen:
   a. Select the option **Create a parameterized filter**.
   b. In the **Package item** box, click the browse button and navigate to the following query item:

      DEFAULT|DEFAULT_DIM|OBJECT_TYPE_DIMENSIONS|

      [SOXBUSENTITY_GPC]| **[SOXBUSENTITY_GPC]|**

      [SOXBUSENTITY_GRANDPARENT]|[GEN_ENTITY_ID]
   c. In the **Operator** box, use **=** (the equal sign is the default).
   d. Clear the box for the **Make this filter optional** option.
   e. If you have multiple queries in your report, in the **Apply filter** box, select the query to which the filter should be applied.
   f. Click **Next**.
5. On the **Populate Control** screen:
   a. Select the **Create new query** option.
   b. In the **Name** box, type a name for the query. For example, **Business Entity Prompt**.
   c. In the **Values to display** box, click the browse button and navigate to the following query item:

      DEFAULT|DEFAULT_DIM|OBJECT_TYPE_DIMENSIONS|

      [SOXBUSENTITY_GPC]| **[SOXBUSENTITY_GPC]|**

      [SOXBUSENTITY_GRANDPARENT]|[GEN_FULL_PATH]
   d. The following values will be displayed in the box:

      [DEFAULT_DIM].[SOXBUSENTITY_GPC].[SOXBUSENTITY_GPC].[SOXBUSENTITY_GRANDPARENT].[GEN_FULL_PATH]
   e. Click **Finish**.
6. Now select the prompt you just created and under the **General** category in the Properties pane, change the **Select UI** value to **List Box**.

Results

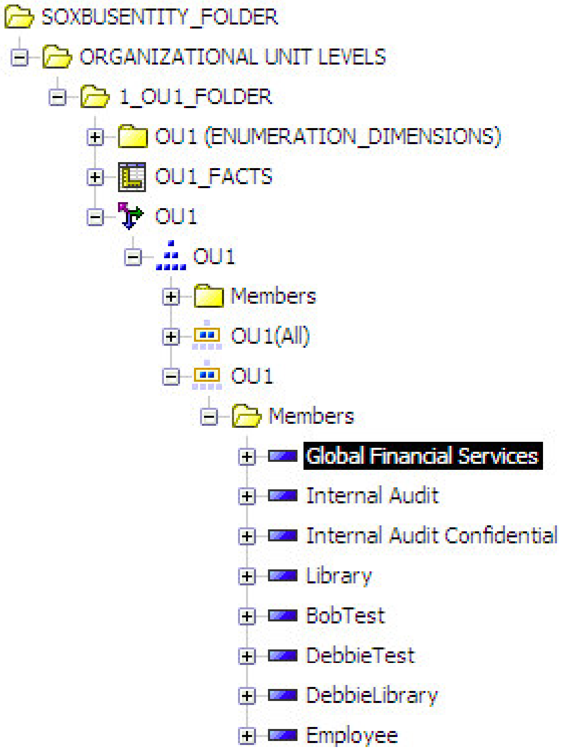
- The SOXBUSENTITY_GPC object type dimension can only be used for filtering purposes. It cannot be used for drill-up/drill-down.
- You will have to adjust the filter created by this prompt to match the objects in your report.

Setting a Default Value

There are instances where you want your prompt to display a default value when the prompt page is displayed to the user.

Procedure

1. Select the Business Entity prompt you created in the previous section "Adding a Single Level Business Entity Dimensional Prompt" on page 95.
2. In the Insertable Objects pane, select the member you would like to set as the default. In this example we are selecting **Global Financial Services**.



3. Right-click on the member and select **Properties**.
4. In the **Properties** dialog, copy the value in the **Member Unique Name**.
5. Click **Close**.
6. Now in the Properties pane for the Business Entity prompt, under **General**, select **Default Selections**.
7. Click the **Add** icon.
8. Paste the value you just copied into the **Add** dialog.
9. Click **Ok**.
10. Click **Ok**.
11. Run the report.

Results

Global Financial Services is now highlighted as the default value in the prompt when you run the report.

Note: The default value you used is environment specific.

Adding an ‘All’ Value

Follow this procedure to allow the user a choice of ‘All’ instead of selecting all of the individual items.

Procedure

1. Select the Business Entity prompt you created in the previous section.
2. In the Insertable Objects pane, select the **OU1(All)** member from the OU1 hierarchy members.
3. Right-click on the member and select **Properties**.
4. In the Properties dialog, copy the value in the **Member Unique Name**.
5. Click **Close**.
6. Now in the Properties pane for the Business Entity prompt, under **Data**, select **Static Choices**.
7. Click the **Add** icon.
8. Paste the value you just copied into the **Use** value in the **Add** dialog. Add **All** to the display value.
9. Click **Ok**.
10. Click **Ok**.
11. Now in the Properties pane for the Business Entity prompt, under **General**, select **Default Selections**.
12. Delete any previous choices.
13. Click the **Add** icon.
14. Paste the value you just copied into the **Add** dialog.
15. Run the report.

Results

The prompt now has an "All" value, and it is selected as the default.

Adding a Tree Prompt

Tree prompts can only be used with the dimensional data model. The allows the user to make a selection from a hierarchical view of the data.

Follow this procedure to add a tree prompt to your report.

Procedure

1. If you do not already have a prompt page in your report, follow the instructions in “Creating a Prompt Page” on page 54.
2. To create your Tree prompt, do the following:
   a. Drag a **Text Item** into the first column of the table. Enter the text for the prompt in the text item.
   b. Select the table cell and set the location to be **Horizontally** right and **Vertically** top.

   c. Make the text bold using the B **Bold** icon.
   d. Drag a Tree Prompt **Tree Prompt** object onto your prompt page. This will bring up the Prompt Wizard.
3. On the Choose Parameter screen of the Prompt Wizard:
   a. Select the option **Create a new parameter**.
   b. In the box, type a name for the parameter. For example, **Business Entity**.
   c. Click **Next**.
4. On the Create Filter screen:
   a. Select the option **Create a parameterized filter**.
   b. In the **Package item** box, click the browse button and navigate to the following hierarchy (object type dimensions will be different on your system):

      DEFAULT_DIM|OBJECT_TYPE_DIMENSIONS|[Business Entity]
   c. In the **Operator** box, use **=** (the equal sign is the default).
   d. Clear the box for the **Make this filter optional** option.
   e. If you have multiple queries in your report, in the **Apply filter** box, select the query to which the filter should be applied.
   f. Click **Next**.
5. On the Populate Control screen:
   a. Select the **Create new query** option.
   b. In the **Name** box, type a name for the query. For example, `Business Entity Prompt`.
   c. Click **Finish**.
6. Now select the prompt you just created and under the **General** category in the Properties pane, change the **Pre-Populate** value to **Yes**.
7. Since the tree prompt can return a large amount of data items you need to set the prompt height to a value that is reasonable.
   a. Select the prompt in the table that you want to set the height.
   b. In the Properties pane, under **Positioning**, select **Size and Overflow**.
   c. In the dialog box, set the height to **1.5 in**.
8. Click **OK**.
9. Using the **Query Explorer**, do the following:
   a. Select the filter that was just created in the main query.
   b. In the Properties pane, under **General**, set the **Application** property to After Auto Aggregation.
10. Run the report.

Results

The tree prompt lets you walk down a data hierarchy and select the exact member you want as a filter.

Note:

- You will have to adjust the filter created by this prompt to match the objects in your report.
- In some instances you will have to create a new member set by creating a data item as follows:

  ```
  [DEFAULT_DIM].[OU1].[OU1]->?Business Entity?
  ```

Filtering on Reporting Period ID

Every report should be filtered by the reporting period ID (either through the use of a prompt or a filter) unless the user wants to report across all reporting periods.

You could, for example, prompt the user explicitly for a specific reporting period and filter by the selected Reporting Period ID. Or, you could filter by the reporting period ID value -1 if the report is only available for the current reporting period.

When combining filtering on Reporting Period ID with filtering on entity, you must use the GEN_REPORTING_PERIOD_ID as the filtering item for the Reporting Period ID.

In order to create a filter on reporting period in a dimensional report, you will use the SOXBusEntity_GPC Object Type Dimension.

Procedure

1. In the Insertable Objects pane, drag and drop the following data item into the detail filters:

   DEFAULT|DEFAULT_DIM|OBJECT_TYPE_DIMENSIONS|
   [SOXBUSENTITY_GPC]| [SOXBUSENTITY_GPC]|
   [SOXBUSENTITY_GRANDPARENT] |GEN_REPORTING_PERIOD_ID]

2. Edit the filter and set the value to -1, the current reporting period, as follows:

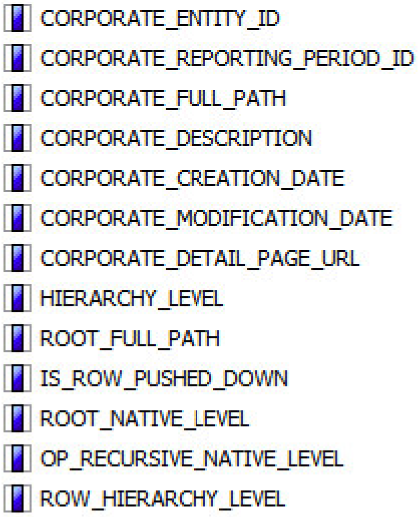
   ```
   [DEFAULT_DIM].[SOXBUSENTITY_GPC].[SOXBUSENTITY_GPC]. [SOXBUSENTITY_GRANDPARENT].
   [GEN_REPORTING_PERIOD_ID] = -1
   ```

3. If you need to set the filter to a prompt value, add the defined prompt value to the expression, as follows:

   ```
   [DEFAULT_DIM].[SOXBUSENTITY_GPC].[SOXBUSENTITY_GPC]. [SOXBUSENTITY_GRANDPARENT].
   [GEN_REPORTING_PERIOD_ID] = ?Reporting Period?
   ```

Results

There is a second method for filtering on the reporting period ID. All object type dimensions have attributes associated with each level of the hierarchy, as shown in Figure 39 on page 102.



CORPORATE_ENTITY_ID
CORPORATE_REPORTING_PERIOD_ID
CORPORATE_FULL_PATH
CORPORATE_DESCRIPTION
CORPORATE_CREATION_DATE
CORPORATE_MODIFICATION_DATE
CORPORATE_DETAIL_PAGE_URL
HIERARCHY_LEVEL
ROOT_FULL_PATH
IS_ROW_PUSHED_DOWN
ROOT_NATIVE_LEVEL
OP_RECURSIVE_NATIVE_LEVEL
ROW_HIERARCHY_LEVEL

Figure 39. Object Type Dimension Attributes

Any one of these attributes can be used in the following expression:

```
roleValue('REPORTING_PERIOD_ID',
[DEFAULT_DIM].[GFS ORG HIERARCHY].[GFS ORG HIERARCHY])
```

This returns the attribute indicated in the expression from any level of the hierarchy. You don't include the level of the hierarchy in the attribute name. You also do not need to worry about the context of the hierarchy.

Working with Data Containers

The following sections describe how to use data containers with dimensional data.

Working with Lists

A list object is used to create a report that will show data in rows and columns When you first decide to use a list object, it will appear empty. This section will explain how to create a dimensional list. We will not repeat the explanation of . For further information on the properties of a list, see "Working with Lists" on page 47.

Selecting Data Columns

At this point you need to select which data columns you want to include in the report and how you want them formatted.

Procedure

1. In the Insertable Objects pane, drag and drop the following data items into the list (what is loaded in your system may be different than what you see here):
   - **DEFAULT|DEFAULT_DIM|OBJECT_TYPE_DIMENSIONS| [GFS ORG HIERARCHY]| [GFS ORG HIERARCHY]| [CORPORATE]**
   - **DEFAULT|DEFAULT_DIM|GRC_OBJECTS|SOXRISK_FOLDER| [SOXRISK]| [SOXRISK]| [SOXRISK]**
   - **DEFAULT|DEFAULT_DIM|GRC_OBJECTS|SOXRISK_FOLDER| [SOXRISK]| [SOXRISK]| [SOXRISK]|[RI_DESCRIPTION]**

- **DEFAULT|DEFAULT_DIM|GRC_OBJECTS|SOXRISK_FOLDER|**

  [SOXRISK]| **[SOXRISK]|** **[SOXRISK]|[RI_FULL_PATH]**

2. In the **Query Explorer**, add the following filter:

   DEFAULT|DEFAULT_DIM|OBJECT_TYPE_DIMENSIONS| **[GFS ORG HIERARCHY]|** **[GFS ORGHIERARCHY]|** **[CORPORATE]|** **[CORPORATE_REPORTING_PERIOD_ID]**
3. Now you need to turn on the drill-up/drill-down behavior:
   a. Select **Data | Drill Behavior** from the top level menu.
   b. In the dialog, click the **Allow drill-up and drill-down** check box.
4. Click **OK**.
5. Now we can set grouping and spanning just like any list report.
   a. Group the business entity dimension and the risk dimension.
   b. Span the risk description and risk full path.

Results

You will also notice that in the OU1 and SOXRISK columns the data is underlined. This indicates that you can drill-up/drill-down on this data. In a list report, the drill-up/drill-down feature acts as an in-line filter. This enables you to focus on the data you want to see more quickly.

In order to drill-down, simply select the underlined data item and the report will drill into the next level of the hierarchy.

Working with Crosstabs

Crosstabs are a type of data container that operates on aggregated data. The row and column fields in the crosstab will be dimensions while the intersection of the crosstab must always be a fact.

You will see three areas in a crosstab object:

- Columns - Data items placed here will generate column headers. Multiple data items can be dropped here to create nested column headers.
- Rows - Data items placed here will generate the rows of data in the crosstab. Multiple data items can be placed here to generate nested rows.
- Measures - Multiple data items can be placed here that will be aggregated against the rows and columns selected.

Selecting Data Rows and Columns

At this point you need to select which data items you want to include as rows and columns in the report and how you want them formatted.

Procedure

1. Go to the **Query Explorer** pane.
2. In the Insertable Objects pane of the drag and drop the following data items into the Data Items pane of the crosstab:
   - **DEFAULT|DEFAULT_DIM|OBJECT_TYPE_DIMENSIONS|**

     [Business Entity]| **[Business Entity]|** **[OU1]**

- **DEFAULT|DEFAULT_DIM|GRC_OBJECTS|SOXCONTROL_FOLDER|SOXCONTROL (ENUMERATION_DIMENSIONS)|**

  [CN_OPERATING_EFFECTIVENES]|

  [CN_OPERATING_EFFECTIVENES]|

  [CN_OPERATING_EFFECTIVENES]

- **DEFAULT|DEFAULT_DIM|GRC_OBJECTS| SOXCONTROL_FOLDER|[SOXCONTROL_FACTS]|[SOXCONTROL (COUNT)]**

Populating with Data

Follow this procedure to populate a crosstab.

Procedure

1. Using the **Page Explorer**, move to **Page 1**.
2. Drag and drop the [CN_OPERATING_EFFECTIVENES] field into the columns area from the **Data Items** tab of the Insertable Objects pane.
3. Drag and drop the [OU1] field into the rows area from the **Data Items** tab of the Insertable Objects pane.
4. Drag and drop the [SOXCONTROL (COUNT)] field into the measures area from the **Data Items** tab of the Insertable Objects pane.
5. Enable drill-up/drill down on the Data|Drill Behavior screen.
6. Run the report.

Results

You will notice that the column and row headers of the crosstab report are underlined. This indicates you can drill-up/drill-down on them. The column headers will act as filters because the operative effectiveness dimension is only a single level.

When you click on Global Financial Services, Global Financial Services appears again because there may be controls directly connected to the Global Financial Service level of the Business Entity hierarchy. When you drill down, the next level must add up to the previous level and that is why Global Financial Services is "Pushed Down" to the next level.

Sorting the Data

Now we want to sort Global Financial Services to the top of crosstab because it is "Pushed Down" from the previous level.

Procedure

1. In the Insertable Objects pane, do the following.
   1. Drag and drop the following data items into the data items pane of the query:

      DEFAULT|DEFAULT_DIM|OBJECT_TYPE_DIMENSIONS|

      [Business Entity]| **[Business Entity]|**

      [OU1]|IS_ROW_PUSHED_DOWN

   b. Change the name of the data item to OU1_IS_ROW_PUSHED_DOWN.
   c. Drag and drop the following data items into the data items pane of the query:

      DEFAULT|DEFAULT_DIM|OBJECT_TYPE_DIMENSIONS|
      [Business Entity]| **[Business Entity]|**
      [OU2]|IS_ROW_PUSHED_DOWN

   d. Change the name of the data item to OU2_IS_ROW_PUSHED_DOWN.
   e. Drag and drop the following data items into the data items pane of the query:

      DEFAULT|DEFAULT_DIM|OBJECT_TYPE_DIMENSIONS|
      [Business Entity]|
      [Business Entity]|
      [OU3]|IS_ROW_PUSHED_DOWN

   f. Change the name of the data item to OU3_IS_ROW_PUSHED_DOWN.
2. On the main report page do the following:
   a. Select the OU1 data item.
   b. In the Properties pane, under **Data**, selecting **Sorting**.
   c. Drag and drop the OU3_IS_PUSHED_DOWN data items to the **Sort List** and select sort descending for each item.
   d. Click **OK**.
   e. Run the report again.
3. Select the **Measures** section of the crosstab.
4. Now click on the Σ **Summation** icon and select **Total** to add summaries to the rows and columns.

Results

For more information on setting crosstab properties, see "Working with Crosstabs" on page 48.

Note:

- You cannot insert two levels from the same hierarchy in a crosstab because the drill-up/drill-down feature will become disabled.
- You can insert multiple measures into a crosstab by dragging both measures to the corner of the crosstab. It will then create multiple measures in the fact cells.

Working with Graphs

Graphs are used to display data in a visual format. There are a variety of graph formats built into IBM Cognos that the report author should become familiar with in order to best display data to the end users. Building a graph with dimensional components enables the end user to be able to drill-up and drill-down on data to better focus the output of the chart.

A graph consists of the following components:

- Default Measure - This is typically the y-axis component of the graph. It will be a numerical value.

- Series - A group or groupings for the y-axis.
- Categories - This is typically the x-axis component of the graph.

Creating a Graph

Follow this procedure to create a graph.

Procedure

1. Create a new Chart report.
   a. Select the **Column** Chart grouping.
   b. Select the **Column with 3D Visual Effect** chart.
   c. Click **OK**.
2. Drag and drop the following query items into the various chart sections:
   - Categories (x-axis)
     DEFAULT|DEFAULT_DIM|GRC_OBJECTS|SOXRISK_FOLDER|SOXRISK (ENUMERATION DIMENSIONS) |
     [RI_RISK_SUBCATEGORY_DEPENDENT_PICKLIST]|
     [RI_RISK_SUBCATEGORY_DEPENDENT_PICKLIST]|
     [RI_RISK_CATEGORY]
   - Default Measure (y-axis)
     DEFAULT|DEFAULT_DIM|GRC_OBJECTS|SOXRISK_FOLDER|[SOXRISK_FACTS]|[SOXRISK (COUNT)]
   - Series
     DEFAULT|DEFAULT_DIM|GRC_OBJECTS|SOXRISK_FOLDER| SOXRISK (ENUMERATION DIMENSIONS)| [RI_RESIDUAL_RISK_RATING]| [RI_RESIDUAL_RISK_RATING]| [RI_RESIDUAL_RISK_RATING]
3. Enable drill-up/drill down on the Data|Drill Behavior screen.
4. Run the report.

What to do next

Work with this graph further by “Adding titles to a Graph.”

Adding titles to a Graph:

Follow this procedure to set some of the titles on the report.

Procedure

1. Open the **Axis Titles** and do the following:
   a. Select the x-axis title.
   b. In the Properties pane, under **General** set the **Default Title** property to **No**.
   c. Double-click on the axis title and set the value to **Risk Category**.
   d. Select the y-axis title.
   e. In the Properties pane, under **General** set the **Default Title** property to **No**.
   f. Double-click on the axis title and set the value to **Risk Count**.
2. Select the **(Default Legend Title)** and do the following:
   a. In the Properties pane under **General**, set the **Default Title** property to **No**.

   b. Double-click on the legend title and set the value to **Residual Risk Rating**.
3. Run the report.

What to do next

Work with this graph further by “Preserving Bar Color.”

Preserving Bar Color:

Now let's examine the drill-up/drill down features. First click on External Fraud.

Since Risk Category is a multi-level dimension, the chart now shows the sub-category data under External Fraud.

if we select Not Determined from the legend, the chart is now filtered even farther to the sub-category and the single residual risk rating. You should also have noticed that the color of the bars was not preserved as we drilled down.

We can fix this by doing the following:

Procedure

1. In the **Query Explorer**, do the following:
   a. Drag a **Data Item** to the Data Items pane.
   b. In the **Data Item**, enter the following formula:
      ```
      caption([RI_RESIDUAL_RISK_RATING])
      ```
   c. In the Properties pane, set the **Name** to **Caption Residual Risk Rating**.
   d. Set the **Aggregate Function** and **Rollup Aggregate Function** to None.
2. In the **Conditional Explorer**, do the following:
   a. Drag and drop a **String Variable** into the Variable pane.
   b. Drag and drop the following data item into the **Expression Definition** pane:
      ```
      [graphMain].[Caption Residual Risk Rating]
      ```
   c. Click Ok.
   d. In the Properties pane, change the **Name** to **Bar Color**.
   e. In the Values pane, add **High**, **Medium**, **Low** and **Not Determined**.
3. On the main report page, do the following:
   a. Select the chart.
   b. In the Properties pane, under **Color and Background**, select **Conditional Palette**.
   c. Select **Bar Color** for the **Variable** value.
   d. Select the **Value** High, click on **Color** and set the color to red.
   e. Select the **Value** Medium, click on **Color** and set the color to yellow.
   f. Select the **Value** Low, click on **Color** and set the color to green.
   g. Select the **Value** Not Determined, click on **Color** and set the color to blue.
   h. Click **OK** to finish.
4. Add the new variable to the properties of the chart:
   a. In the main report page, click on RI_RESIDUAL_RISK_RATING.
   b. Using the **Ancestor** selector, select **Chart Node Member**.
   c. In the Properties pane, under **Data**, select **Properties**.

   d. Check the **Caption Residual Risk Rating**.
5. Select the new crosstab container and run the report.

What to do next

Work with this graph further by “Preserving Legend Colors.”

Preserving Legend Colors:

The colors of the bars on the graph have been preserved as we drill down but the legend no longer matches the bar colors. This is because we used a conditional palette to force the bar colors to specific values and the legend will not pick up those values.

In order to fix the legend we need to create our own legend. Do the following:

Procedure

1. Click the **Toolbox** tab on the Insertable Objects pane.
2. Drag a Table **Table** object into the body of the report page. Select 2 columns by 1 row.
3. Drag the existing chart into the left hand column of the table.
4. Drag a Crosstab **Crosstab** into the second column of the table.
5. In the Properties pane, under **Data**, set the **Query** property to be the same as the existing chart.
6. Populate the crosstab as follows:
   a. From the Insertable Objects pane, drag and drop **[RI_RESIDUAL_RISK_RATING]** into the rows of the crosstab.
   b. From the Insertable Objects pane, drag and drop **[RI_RESIDUAL_RISK_RATING]** into the columns of the crosstab.
   c. From the Insertable Objects pane, drag and drop **[SOXRISK (COUNT)]** into the measures of the crosstab.
7. Configure the columns and measures as follows:
   a. In the Properties pane, under **Box**, set the **Box Type** to **None** for the columns and the measures.
   b. Remove the borders from the remaining cells of the crosstab.
   c. Set the **Background Color** to **White** for the remaining cells of the crosstab.
8. Select the text in the crosstab corner and do the following:
   a. In the Properties pane, under **Text Source**, set the **Source Type** to Text.
   b. Set the **Text** property to Residual Risk Rating.
   c. Set the **Horizontal Alignment** to **Left**.
   d. Set the **Vertical Alignment** to **Bottom**.
9. Add the new variable to the properties of the legend crosstab:
   a. In the main report page, click on **RI_RESIDUAL_RISK_RATING**.
   b. Using the **Ancestor** selector, select **Crosstab Node Member**.
   c. In the Properties pane, under **Data**, select **Properties**.
   d. Check the **Caption Residual Risk Rating**.
10. In the main report page, click on **RI_RESIDUAL_RISK_RATING** and do the following:
    a. In the Properties pane, under **Box**, select **Padding**.

   b. Set the left padding to 5px.
   c. Click **Ok**.

11. In order to create the color boxes for the legend, do the following:
    a. Unlock the report template.
    b. Drag and drop a ab Text Item **Text Item** to the left of the text item **[RI_RESIDUAL_RISK_RATING]** in the legend crosstab.
    c. Enter 0 as the text.
    d. Set the font size to 2 pt.
    e. In the Properties pane, under **Positioning**, set the **Size & Overflow** to 10px by 10px.
    f. In the Properties pane, under **Conditional**, set the **Style Variable** to **Bar Color.**
    g. In the **Conditional Explorer**, select **High** from the **Bar Color** variable.
    h. Set the **Background Color**, **Foreground Color** and **Border Color** to **Red**.
    i. Repeat the above steps for medium (Yellow), low (Green) and Not Determined as choices. The report template needs to remain unlocked.
12. In the table cell that holds the legend crosstab, do the following:
    a. Set the horizontal alignment to left.
    b. Set the vertical alignment to top.
13. In the table cell that holds the original crosstab, do the following:
    a. Set the horizontal alignment to right.
    b. Set the vertical alignment to top.
14. In the table that holds the two legends, in the Properties pane, under **Positioning**, set the **Size & Overflow** property to blank.
15. In the main chart, select the legend and do the following:
    a. In the Properties pane, under **Box**, set the **Box Type** to None.
16. Run the report.

Results

You now see the custom legend has the same colors as the chart. Due to the fact that it is a crosstab, the items in the legend are underlined indicating that you can drill-up and drill-down on them. Since the legend and the chart share the same underlying query, the chart will behave according to how you drill-up/drill-down in the legend.

Tip: Creating the custom legend is a workaround that enables us to pick up colors from the conditional palette.

Filtering Data

To filter data in a dimensional report, you can only use fields from the dimensional model. As a report author you have access to both the relational and dimensional models but including fields from the relational model in a dimensional report will generate unpredictable SQL and is not recommended.

Using the SOXBusEntity_GPC Dimension

In order to facilitate the creation of dimensional reports, a dimension called the SOXBUSENTITY_GPC dimension has been added by default to the

OBJECT_TYPE_DIMENSIONS folder.

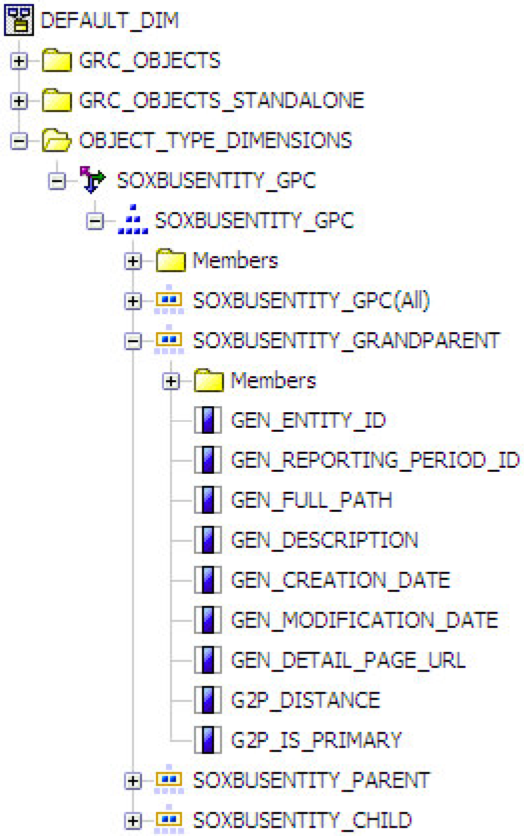


Figure 40. SOXBUSENTITY_GPC Dimension

This object can only be used for filtering. It cannot be used to drill-up or drill-down.

The fields in the dimension are identical to those in the relational model and enable you to create reporting period filters, business entity filters, distance filters and full path filters in a dimensional report.

Using a Context Filter

A context filter is used to limit a dimensional report to the context of a particular member of a hierarchy.

Follow this procedure to create a context filter.

Procedure

1. Open an existing dimensional report.
2. Select the query you would like to filter from the **Query Explorer**.
3. Expand the object model view in the Insertable Objects pane to show the members of the dimension you would like to filter on.
4. Drag and drop the member you would like to filter on into the context filter area.

Results

There is now a fixed filter on this report limiting the initial output from the OU1 dimension to Global Financial Services. Since this report is dimensional, you can still drill-up/drill-down on Global Financial Services.

Creating Drill-Throughs

In CommandCenter there are many more data items that can be used for drill-throughs. This opens up possibilities of attaching different sub-reports to many different objects in a report such as the legend.

When you are creating a dimensional report you can drill-through either to another dimensional report or a relational report. The techniques for doing this are vastly different.

Creating a Dimensional to Dimensional Drill-Through

In order to demonstrate the technique for creating the a dimensional drill-through, create a simple dimensional list report.

Procedure

1. Create a new list report.
2. In the Insertable Objects pane, drag and drop the following data items into the list (please remember that what is loaded in your system will be different then what you see here):
   - **DEFAULT|DEFAULT_DIM|OBJECT_TYPE_DIMENSIONS|[Business Entity]|[Business Entity]|[Business Entity]**
   - **DEFAULT|DEFAULT_DIM|GRC_OBJECTS|SOXCONTROL_FOLDER|[SOXCONTROL]|[SOXCONTROL]|[SOXCONTROL]**
   - **DEFAULT|DEFAULT_DIM|GRC_OBJECTS|SOXCONTROL_FOLDER|[SOXCONTROL]|[SOXCONTROL]|[SOXCONTROL]|[CN_DESCRIPTION]**
   - **DEFAULT|DEFAULT_DIM|GRC_OBJECTS|SOXCONTROL_FOLDER|SOXCONTROL (ENUMERATION_DIMENSIONS)|[CN_OPERATING_EFFECTIVENES]|[CN_OPERATING_EFFECTIVENES]|[CN_OPERATING_EFFECTIVENES]**
3. Cut the `[CN_OPERATING_EFFECTIVENES] from the list report. This is being removed because having enumerated strings in a dimensional list report is not supported.`
4. Go to the **Query Explorer** and add the following filters to the query:

```
[DEFAULT_DIM].[BUSINESS ENTITY].[BUSINESS ENTITY] = ?Business Entity?
```

```
[DEFAULT_DIM].[CN_OPERATING_EFFECTIVENES].
[CN_OPERATING_EFFECTIVENES].[CN_OPERATING_EFFECTIVENES] = ?Operating
Effectiveness?
```

```
[DEFAULT_DIM].[SOXBUSENTITY_GPC].[SOXBUSENTITY_GPC].
[SOXBUSENTITY_GRANDPARENT].[GEN_REPORTING_PERIOD_ID] = -1
```

5. On the first two filters, in the Properties pane, under **General**, set the **Application** property to After Auto-Aggregation.
6. Enable drill-up/drill down on the **Data|Drill Behavior** screen.
7. Save the report as **Dimensional Sub-Report**.
8. Using the list report you created "Working with Crosstabs" on page 103, make a copy and do the following:
   a. Select the main page of the report.
   b. Unlock the crosstab.
   c. Select the measure in the crosstab.
   d. In the Properties pane, under **Data**, select **Drill-Through Definitions**.
   e. Add a new drill-through definition.
   f. Set the name of the drill-through to **Dimensional Sub-Report**.
   g. Select the report **Dimensional Sub-Repor**t.
   h. Check to open the report in a new window.
   i. Edit the report parameters and select **OU1** for the **Business Entity** parameter and **CN_OPERATING_EFFECTIVENES** for the **Operating Effectiveness** parameter. Set the **Property to Pass** value to **Member Unique Name** for both parameters.
9. Run the report.

Results

The measures in the crosstabs now appear blue and underlined indicating a drill-through is attached to that cell.

Adding a Dimensional to Relational Drill-Through

In order to demonstrate the technique for creating a relational drill-through, we are going to create a simple relational list report.

Procedure

1. Create a new list report.
2. In the Insertable Objects pane, drag and drop the following data items into the list (please remember that what is loaded in your system will be different then what you see here):
   - **DEFAULT|DEFAULT_REL|GRC_OBJECTS| [SOXBUSENTITY_FOLDER| [SOXBUSENTITY_GPC]|[CEN_NAME00]**
   - **DEFAULT|DEFAULT_REL|GRC_OBJECTS| [SOXCONTOL]|[CN_NAME00]**
   - **DEFAULT|DEFAULT_REL|GRC_OBJECTS| [SOXCONTOL]|[CN_DESCRIPTION]**
   - **DEFAULT|DEFAULT_REL|GRC_OBJECTS| [SOXCONTOL]|ENUMERATION_FIELDS| OPERATING_EFFECTIVENES (ENUMERATION)|[CN_OPERATING_EFFECTIVENES]**
3. Go to the **Query Explorer** and add the following filters to the query:

```
[DEFAULT_REL].[SOXBUSENTITY_GPC].[GEN_ENTITY_ID] = ?Business Entity?
```

```
[DEFAULT_REL].[SOXCONTROL].[CN_OPERATING_EFFECTIVENES_ID] = ?Operating
Effectiveness?
[DEFAULT_REL].[SOXBUSENTITY_GPC].[GEN_REPORTING_PERIOD_ID] = -1
```

4. Save the report as **Relational Sub-Report**.
5. Using the list report you created "Working with Lists" on page 102, make a copy and do the following:
   a. Drag and drop a **Data Item** into the Data Items pane.
   b. Add the following expression:
   ```
   roleValue('_businessKey',[DEFAULT_DIM].[BUSINESS ENTITY]. [BUSINESS ENTITY])
   ```
   c. In the Properties pane, under **Data Item**, set the **Name** to Business Entity ID.
   d. Set the **Aggregate Function** and Rollup **Aggregate Function** to none.
   e. Drag and drop a **Data Item** into the Data Items pane.
   f. Add the following expression:
   ```
   roleValue('_businessKey',[CN_OPERATING_EFFECTIVENES])
   ```
   g. In the Properties pane, under **Data Item**, set the **Name** to Operating Effectiveness ID.
   h. Set the **Aggregate Function** and Rollup **Aggregate Function** to none.
6. On the main page of the report, do the following:
   a. Select the measure in the crosstab.
   b. In the Properties pane, under **Data**, select **Drill-Through Definitions**.
   c. Add a new drill-through definition.
   d. Set the name of the drill-through to **Relational Sub-Report**.
   e. Select the report **Relational Sub-Report**.
   f. Check to open the report in a new window.
   g. Edit the report parameters and select **Business Entity ID** for the **Business Entity parameter** and **Operating Effectiveness ID** for the **Operating Effectiveness parameter**.
7. Add the two variables to the properties of the list:
   a. In the main report page, select the list.
   b. In the Properties pane, under **Data**, select **Properties**.
   c. Check the **Business Entity Id** and the **Operating Effectiveness ID**.
8. Run the report.

Results

The measures in the crosstabs now appear blue and underlined indicating a drill-through is attached to that cell.

Creating Joins to Secondary Objects

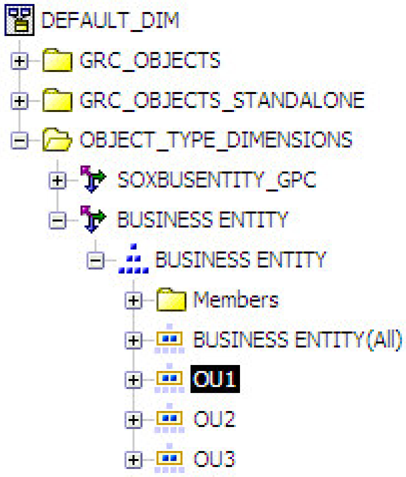
In the framework there are no default joins to secondary objects. In the relational model you can create a join using the join object in the Query Explorer. In the dimensional model you cannot do this.

To solve this problem in the dimensional model, you can create a join to a secondary object by creating specific definitions in IBM OpenPages GRC on Cloud.

As an example, create a join between Business Entities and Issues in the dimensional model.

Procedure

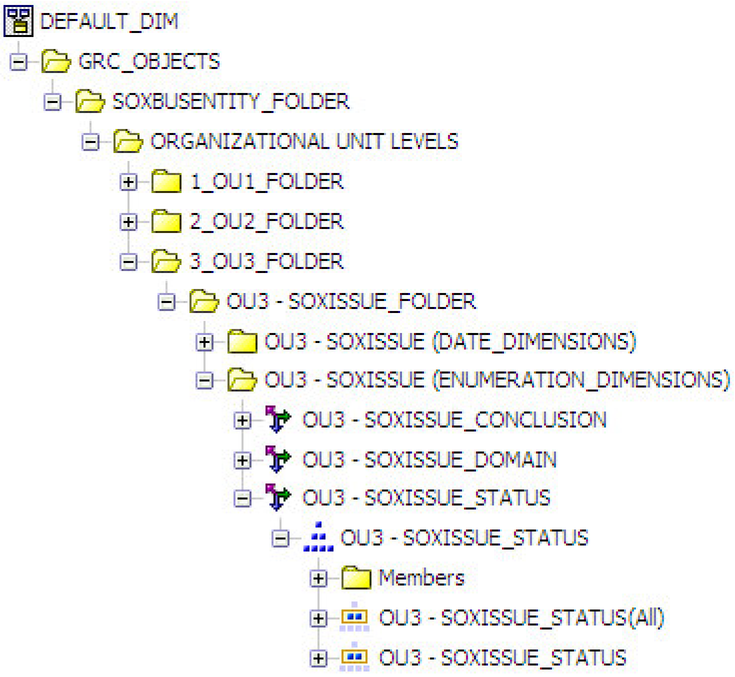
1. You must first create a Recursive Object Level Set in the SOXBusEntity object:
   a. From the main menu go to **Administration** | **Object Types**.
   b. Select **SOXBusEntity** from the list.
   c. Scroll to the bottom to **Recursive Object Levels** and click **Edit.**
   d. Save the object.
2. Now you need to create an Object Type Dimension:
   a. From the main menu, go to **Administration** | **Reporting Framework** | **Configuration**.
   b. In **Object Type Dimensions**, click **Add**.
   c. For the **Name**, enter **Entity-Issue**.
   d. Select a **Starting Object Type** of **OU1** and click **GO**.
   e. Click **Choose Object Type** next to **OU3** and select **Issue**.
   f. Click **Save**.
3. Regenerate the framework.
4. Create a new crosstab report.
5. From **Object Type Dimensions** add the **OU1** level from the existing Business Entity Object Type Dimension to the rows of the crosstab.



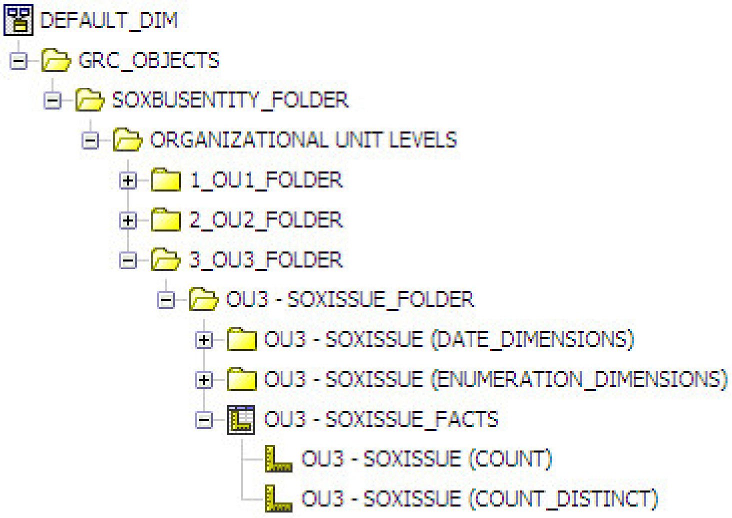
DEFAULT_DIM
- GRC_OBJECTS
- GRC_OBJECTS_STANDALONE
- OBJECT_TYPE_DIMENSIONS
  - SOXBUSENTITY_GPC
  - BUSINESS ENTITY
    - BUSINESS ENTITY
      - Members
      - BUSINESS ENTITY(All)
      - OU1
      - OU2
      - OU3

6. From the **GRC_OBJECTS SOXBUSENTITY** folder, add the **OU3 - SOXISSUE_STATUS** to the column cell of the crosstab. This new enumeration is the result of the join created in the framework due to the definition of the

Entity-Issue Object Type Dimension.



7. From the **GRC_OBJECTS SOXBUSENTITY** folder, add the **OU3 - SOXISSUE (COUNT)** fact to the measure cell of the crosstab. This new fact is the result of the join created in the framework due to the definition of the Entity-Issue Object Type Dimension.



8. In the **Query Explorer**, add the following reporting period filter:

```
[DEFAULT_DIM].[BUSINESS ENTITY].[BUSINESS ENTITY].[OU1].
[OU1_REPORTING_PERIOD_ID] = -1
```

9. Enable drill-up/drill down on the Data|Drill Behavior screen.
10. Run the report.

Results

For more information on creating Recursive Object Levels and Object Type Dimensions please see the *IBM OpenPages GRC on Cloud Administrator's Guide*.

Changing 'All' Level Titles

In the dimensional reports that you create, each dimension has a level above the top level of the hierarchy which is referred to as the "All" level. You cannot remove this level, it is automatically generated by the framework.

The problem with this level is that the text string that is generated for this level is based upon the name of the dimension. So for instance, under a Loss Event object if you have elected to have a dependent picklist generated as a hierarchical dimension, the name would be LE_RISK_SUBCATEGORY_DEPENDENT_PICKLIST (All) (in RDL) or Loss Event Risk Sub-Category Dependent Picklist (All) in English. Other names, such as RI_RISK_CATEGORY (All), would turn into Risk Risk Category (All) in English. Neither one of these is desirable on a report. We will now describe two methods for solving this problem.

Setting the 'All' Text in a Report

At the report level you can create formulas to correct this problem.

For example, in the Loss Event Trend Report the occurrence date All text is "Loss Event Occurrence Start Date (All)" and the Risk Category All text is "Loss Event Risk Subcategory (All)".

Procedure

1. In the **Query Explorer**, do the following:
   a. Drag and drop a Data Item **Data Item** into the Data Items pane.
   b. Add the following expression:
      ```
      if (roleValue('_businessKey',[ORM1_DIM].
      [LE_RISK_SUBCATEGORY_DEPENDENT_PICKLIST].
        [LE_RISK_SUBCATEGORY_DEPENDENT_PICKLIST]) is missing )
          then (1)
          else (2)
      ```
   c. In the Properties pane, under **Data Item**, set the **Name** property to **Risk Category Title**.
   d. Set the **Aggregate Function** and **Rollup Aggregate Function** to **None**.
2. In the **Conditional Explorer**, do the following:
   a. Create a string variable with the following expression:
      ```
      [graphMain].[Risk Category Title]
      ```
   b. Add a value called 1.
3. Now on the main report page, do the following:
   a. Select LE_RISK_CATEGORY in the **Series**.
   b. In the Properties pane, under **Conditional**, set the **Text Source Variable** to **Risk Category Title**.
4. Select LE_RISK_CATEGORY in the **Series**.
5. In the **Conditional Explorer**, do the following:
   a. Select the 1 value from the Risk Category Tile variable.
   b. Select LE_RISK_CATEGORY in the **Series**.
   c. In the Properties pane, under **Text Source**, set the **Text** value to **All**.
6. Add the new variable to the properties of the graph series:
   a. In the main report page, click on LE_RISK_CATEGORY.
   b. Using the **Ancestor** selector, select **Chart Node Member**.

c. In the Properties pane, under **Data**, select **Properties**.

d. Check the **Risk Category Title**.

7. Run the report.

   The risk category now shows the text "All" instead of the default text. We can do the same for the occurrence date drill up value.

   Do the following:

8. In the **Query Explorer**, do the following:

   a. Drag and drop a **Data Item** into the Data Items pane.

   b. Add the following expression:

   ```
   if (roleValue('_businessKey',[ORM1_DIM].[LE_OCCURRENCE_DATE Y-Q-M].
     [LE_OCCURRENCE_DATE Y-Q-M]) is missing)
       then (1)
       else (2)
   ```

   c. In the Properties pane, under **Data Item**, set the **Name** property to **Occurrence Date Title**.

   d. Set the **Aggregate Function** and **Rollup Aggregate Function** to **None.**

9. In the **Conditional Explorer**, do the following:

   a. Create a string variable with the following expression:

   ```
   [graphMain].[Occurrence Date Title]
   ```

   b. Add a value called 1.

10. Now on the main report page, do the following:

    a. Select YEAR_KEY in the **Categories**.

    b. In the Properties pane, under **Conditional**, set the **Text Source Variable** to **Occurrence Date Title**.

11. Select YEAR_KEY in the **Categories**.

12. In the **Conditional Explorer,** do the following:

    a. Select the 1 value from the Occurrence Date Tile variable.

    b. In the Properties pane, under **Text Source**, set the **Text** value to **All Occurrence Dates.**

13. Add the new variable to the properties of the graph series:

    a. In the main report page, click on YEAR_KEY.

    b. Using the **Ancestor** selector, select **Chart Node Member**.

    c. In the Properties pane, under **Data**, select **Properties**.

    d. Check the **Occurrence Date Title**.

14. Run the report.

Results

The occurrence date now shows the text "All Occurrence Dates" instead of the default text.

Setting the Global 'All' Value

Setting the All value on a report by report basis consumes a lot of time. You also have the option of setting the All value for the entire set of dimensions in your framework. The limitation on this method is that it forces every value to be the same value.

Procedure

1. Go to **Administration | Application Text** from the main menu.

2. Under the folder **Reporting Framework**, go to the setting **com.fw.template.dimension.all.**
3. Change the value from {0} (All) to **All** for each language you are using.
4. Regenerate the framework.

Using Data Labels

In some of the previous examples you might have noticed that the titles have been fixed as a text item. This might cause issues as you drill-up/drill-down because you lose context of where you are in the hierarchy.

Follow this procedure to change titles in an example report.

Procedure

1. Select the Risk Category text item that serves as the name of the Legend and do the following:
   a. In the Properties pane, under **Text Source**, set the **Source Type** to D**ata Item Label**.
   b. Set the **Data Item Label** property to **LE_RISK_CATEGORY**.
2. Run the report.

Results

You can now see that the legend title has been replaced with the label of the parent category that you drilled from.

Chapter 9. Working with Business Entities

A business entity is a hierarchical object. Business entities can have parents and children which are themselves business entities. Any level of the business entity structure can have other objects linked to them such as processes and risk assessments.

By using business entities you can perform the following:

- Scope and filter report data
- Determine how data instances are counted
- Determine how numbers are displayed (such as, individual or rollup numbers for a selected business entity)
- Enable drill-down/drill-up in reporting

There are two views of business entities: GPC (grandparent-parent-child) which exists in the framework by default and a hierarchical user defined model.

The user defined model can be created in IBM OpenPages GRC on Cloud. Please see the *IBM OpenPages GRC on Cloud Administrator's Guide* for more information on setting up these recursive object levels and object type dimensions for the dimensional model for a business entity.

The SOXBUSENTITY_GPC Query Subject

The SOXBUSENTITY_GPC query subject is the most basic entity object. It is similar to the GPC component object except that all the values are part of the same query subject.

The query subject has three abstracted levels of business entities that you can use in your report:

- Grandparent entity (GEN)
- Parent entity (PEN)
- Child entity (CEN)

In this model, only the child entity has other objects associated with it.

This query subject is intended to have backwards compatibility. If you have reports written against the legacy framework, they can use this query subject to replace the previous SOXBUSENTITY object. The only difference between this query subject and the previous version, is that security has been applied to the grandparent level.

This is the primary query subject you will be using when working with business entities in the relational model.

The User Defined Entity Hierarchy

In the user defined model, the user defines the recursive object levels and the names of the levels. By doing this, once the framework is regenerated, you will have objects in your model with the names you selected in the UI. For instance, if your hierarchy consists of regions, countries, states, towns you will see four folders in your framework with these names, query subjects and the appropriate relationships between them.

Note: You must configure your system to include the entire depth of your hierarchy or the framework model created will not function properly.

In addition to defining the names of the levels of your business entity hierarchy, you can define an object type dimension which creates a hierarchical object in the dimensional framework. This enables you to use dimensional models in lists, crosstabs and charts. Only by using dimensional models can you use drill-up/drill-down features.

When a user defined hierarchy is created, one query subject is created for each level in the hierarchy. These are advantages and disadvantages of this object structure:

- Each level of the hierarchy is pre-filtered and will only return entities at that level.
- These objects are ideal for creating cascaded entity level prompts.
- They can be used as individual fields on a list report without having to create complex filters.
- The field headers will reflect the names you selected when you defined the hierarchy.

These are the primary objects you will be using when working with business entities in the dimensional model.

GPC Component Objects

The GPC Component Objects are included by default in your model. This object can only be found in the **Miscellaneous | Advanced | GPC Component Objects folder**. Three query subjects created are equivalent to a grandparent, parent and child. When using these you need to be aware of the following:

- If you use the query subjects standalone, every entity will appear at the grandparent, parent and child levels.
- If you use all three levels of the GPC in a report, you must add filters to obtain the correct values at each level.
- You can use the grandparent level to create a prompt that contains entities from all levels of the hierarchy.
- Reports written with the GPC objects are portable between dissimilar environments.

It is recommended that reports be written with the SOXBUSENTITY_GPC query subject instead of this model.

The GPC Object Type Dimension

In the dimensional model, a default dimension is created called SOXBUSENTITY_GPC. This dimension is hierarchical with three levels; grandparent, parent and child. Each level in the hierarchy has only the system attributes of an entity object defined. This dimension is only used for filtering in a dimensional report and cannot be used as a dimension itself which explains why only the system attributes are defined.

The GPC Data Model

There are three levels of entities in the GPC object model: grandparent, parent, and child. This forms an abstraction of your data. Even if you have more than three levels in your entity hierarchy, this model can be used.

You can view these three levels as follows:

- Grandparents have no parents, but can have children.
- Parents can have parents, can have children and can be children.
- Children can have parents.

The numeric columns in the report represent the distance between entities in the entity hierarchy. An entity is always a parent of itself and a parent is always a child of itself. This can be seen where the distance equals zero.

Table 43. Typical entity level usage

Use this level...	For...
grandparent (GEN_)	Filtering and scoping
parent (PEN_)	Grouping and aggregating data, and prompting
child (CEN_)	Filtering and scoping

Filtering Relational Business Entities

The following topics describe how to configure filtering in reports.

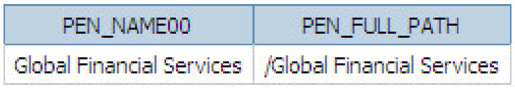
- "Filtering Using the Full Path"
- "Filtering Hierarchy Depth" on page 124
- "Filtering Using Cascaded Prompts" on page 124
- "Quick Reference for Displaying and Filtering Data" on page 127

Filtering Using the Full Path

The full path variable that exists on all business entity objects is commonly used to allow the user to filter the context of the report. The directions for creating this type of prompt can be found on "Adding a Business Entity Prompt" on page 56. Once the prompt is created, you can adjust the filter that the prompt wizard creates to achieve the results you want.

In this first example, we are going to display the parent name and full path.

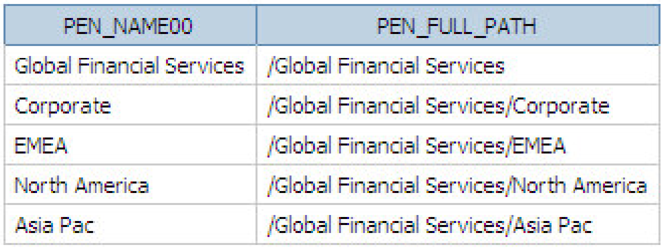
What this filter is doing is only allowing the user to see the exact entity that was chosen in the prompt. The second filter indicates that you only want to see entities that have distance of zero from the selected entity, as shown in Figure 41 on page 122. This filter can only return the entity that was chosen.



PEN_NAME00	PEN_FULL_PATH
Global Financial Services	/Global Financial Services

Figure 41. Business Entity Filtering - G2P = 0

If I eliminate the second filter, I obtain the following, as shown in Figure 42:

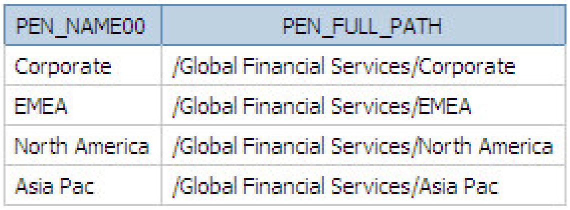


PEN_NAME00	PEN_FULL_PATH
Global Financial Services	/Global Financial Services
Corporate	/Global Financial Services/Corporate
EMEA	/Global Financial Services/EMEA
North America	/Global Financial Services/North America
Asia Pac	/Global Financial Services/Asia Pac

Figure 42. Business Entity Filtering - No G2P Filtering

This shows the entity selected in the prompt and the direct children below that entity.

If I change the G2P filter to 1 (show parents that are one join away from the grandparent), I obtain the following, as shown in Figure 43:

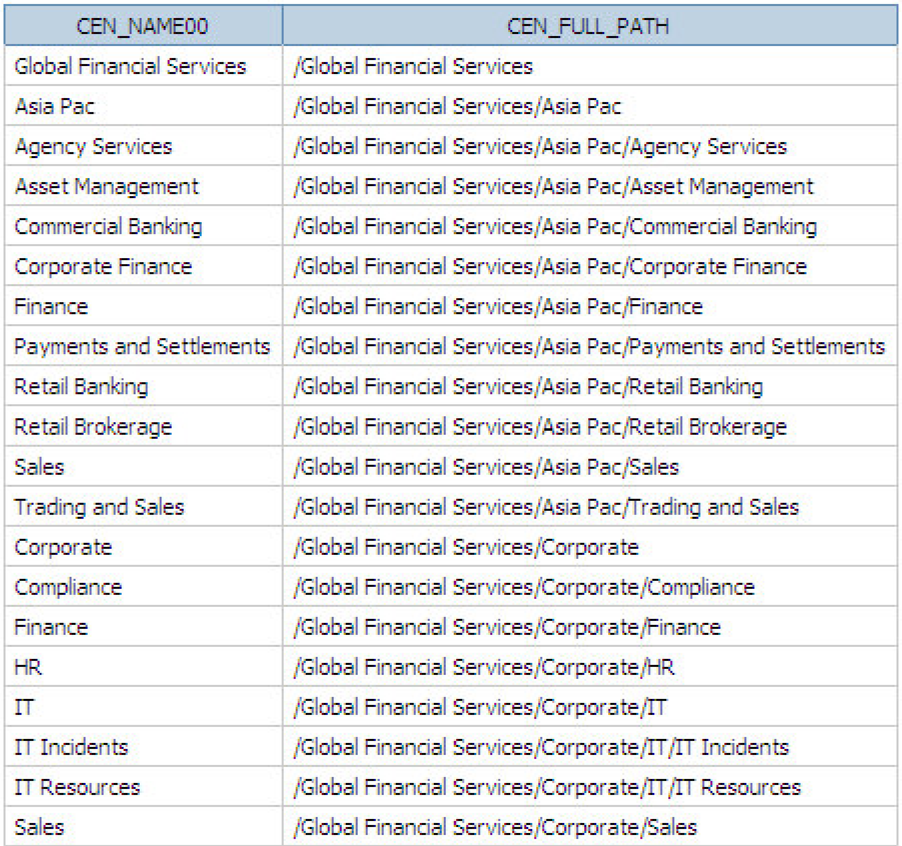


PEN_NAME00	PEN_FULL_PATH
Corporate	/Global Financial Services/Corporate
EMEA	/Global Financial Services/EMEA
North America	/Global Financial Services/North America
Asia Pac	/Global Financial Services/Asia Pac

Figure 43. Business Entity Filtering - G2P = 1

This shows only the children of the entity selected.

Now if I change the display variable to the child entity and remove the distance filter, I obtain the following when I run the report, as shown in Figure 44 on page 123:



CEN_NAME00	CEN_FULL_PATH
Global Financial Services	/Global Financial Services
Asia Pac	/Global Financial Services/Asia Pac
Agency Services	/Global Financial Services/Asia Pac/Agency Services
Asset Management	/Global Financial Services/Asia Pac/Asset Management
Commercial Banking	/Global Financial Services/Asia Pac/Commercial Banking
Corporate Finance	/Global Financial Services/Asia Pac/Corporate Finance
Finance	/Global Financial Services/Asia Pac/Finance
Payments and Settlements	/Global Financial Services/Asia Pac/Payments and Settlements
Retail Banking	/Global Financial Services/Asia Pac/Retail Banking
Retail Brokerage	/Global Financial Services/Asia Pac/Retail Brokerage
Sales	/Global Financial Services/Asia Pac/Sales
Trading and Sales	/Global Financial Services/Asia Pac/Trading and Sales
Corporate	/Global Financial Services/Corporate
Compliance	/Global Financial Services/Corporate/Compliance
Finance	/Global Financial Services/Corporate/Finance
HR	/Global Financial Services/Corporate/HR
IT	/Global Financial Services/Corporate/IT
IT Incidents	/Global Financial Services/Corporate/IT/IT Incidents
IT Resources	/Global Financial Services/Corporate/IT/IT Resources
Sales	/Global Financial Services/Corporate/Sales

Figure 44. Business Entity Filtering - All Levels

What you now see (only part of the data is shown) is the entity selected and all entities below what was selected. This is useful when you want to create a roll-up of all information below a selected entity.

What these examples show is the flexibility you have when filtering on the business entity object to obtain data on a specific entity, an entities children or everything rolled up to a business entity.

Relational Entity Context Fields

Table 44 is an explanation of the context fields used to filter entities in the relational model.

Table 44. Relational Entity Context Fields

Field Name	Description
G2P_DISTANCE	The number of levels between a grandparent and a parent. The value is either 0 or 1.
G2P_IS_PRIMARY	Indicates whether the grandparent object is the primary parent of the parent object. Values are "Y" and "N".
P2C_DISTANCE	The number of levels between a parent and a child. The values could be 0 to N.
P2C_IS_PRIMARY	Indicates whether the parent object is primary parent of the child object. Values are "Y" and "N".

Filtering Hierarchy Depth

Another method for filtering the business entity is to create a prompt based upon the depth of an entity in the hierarchy.

Note: If you have an IBM DB2 database, you must replace the Oracle `INSTR` function in the following sample expression with the corresponding DB2 function.

For instance, let’s assume you have a uniformly distributed business entity hierarchy and wanted the user’s to only pick from the third level of the hierarchy. You could place the following filter on your prompt so they can only see the third level of the hierarchy:

```
instr([PEN_FULL_PATH],'/',1,3) > 0 and
instr([PEN_FULL_PATH],'/',1,4) = 0
```

This filter uses the number of forward slashes in the full path variable to determine the absolute depth of an entity in the hierarchy.

All of the items have 3 slashes in them which is what the filter was designed to display.

Filtering Using Cascaded Prompts

Cascaded prompts are used when the user wants to only see prompt values based upon a selection of a previous prompt. This example will show you how to use the objects in the GPC folders to create a series of cascaded prompts. Do the following:

Procedure

1. Create a new list report.
2. Follow the instructions in “Creating a Prompt Page” on page 54.
3. First we will create a global level prompt, do the following:
   a. Drag a ab Text Item **Text Item** into the first column of the table. Enter the text for the prompt in the text item.
   b. Select the table cell and set the location to be **Horizontally** right and **Vertically** top.
   c. Make the text bold using the **B** **Bold** icon.
   d. Drag a Value Prompt **Value Prompt** object into the second column on your prompt page. This will bring up the Prompt Wizard.
4. On the **Choose Parameter** screen of the Prompt Wizard:
   a. Select the option **Create a new parameter**.
   b. In the box, type a name for the parameter. For example, **Global Entity**.
   c. Click **Next**.
5. On the **Create Filter** screen:
   a. Select the option **Create a parameterized filter**.
   b. In the **Package item** box, click the browse button and navigate to the following query item (the names of these query items are user defined, so they may be different on your system):

      ```
      DEFAULT_REL|GRC_OBJECTS|SOXBUSENTITY_FOLDER|REGIONAL HIERARCHY|
      1_GLOBAL_FOLDER|ID_FIELDS|[GLOBAL]|ID_FIELDS|[GLOBAL_ENTITY_ID]
      ```

   c. In the **Operator** box, use **=** (the equal sign is the default).
   d. Clear the box for the **Make this filter optional** option.

   e. If you have multiple queries in your report, in the **Apply filter** box, select the query to which the filter should be applied.
   f. Click **Next**.
6. On the **Populate Control** screen:
   a. Select the **Create new query** option.
   b. In the **Name** box, type a name for the query. For example, **Global Entity Prompt**.
   c. In the **Values to display** box, click the browse button and navigate to the following query item:
      ```
      DEFAULT_REL|GRC_OBJECTS|SOXBUSENTITY_FOLDER|REGIONAL HIERARCHY|
      1_GLOBAL_FOLDER|[GLOBAL]|[GLOBAL_NAME00]
      ```
   d. The following values will be displayed in the box:
      ```
      [DEFAULT_REL].[GLOBAL].[GLOBAL_NAME00]
      ```
   e. Click **Finish**.
7. Now select the prompt you just created and under the **General** category in the Properties pane, change the **Select UI** value to **List Box**.
8. In the Properties pane under **Data**, set the **Sorting** property to be the name of the business entity.
9. If there are a limited number of items in the prompt, you can set the prompt height to a value that shows just the prompts and no white space.
   a. Select the prompt in the table that you want to set the height.
   b. In the Properties pane, under **Positioning**, select **Size and Overflow**.
   c. In the dialog box, set the height to **14 pt** times the number of rows in the prompt.
10. Now we need to add a reprompt button to the prompt page. Do the following:
   a. Drag a Prompt Button **Prompt Button** into the table cell below the value prompt you just created.
   b. In the Properties pane under **General**, set the **Type** property to **Reprompt**.
   c. Drag a Text Item **Text Item** on top of the prompt button. Enter the following text: **Update Regional Entity**.
   d. In the Proprieties pane under **Color & Background**, set the **Background Color** property to **Silver**.
   e. Select the table cell that the button has been placed in.
   f. In the Properties pane under **Positioning**, set the **Size & Overflow** property to **40 px**.
11. In the **Query Explorer**, select the **Global Entity Prompt** query subject and do the following:
   a. Click the **Toolbox** tab on the Insertable Objects pane.
   b. Drag and drop a Filter **Filter** object into the Filter pane of the query.
   c. In the Expression Definition pane enter the following:
      ```
      [DEFAULT_REL].[GLOBAL].[IS_ROW_PUSHED_DOWN]='N'
      ```
      The data item can be found in the following folder:
      ```
      DEFAULT_REL|GRC_OBJECTS|SOXBUSENTITY_FOLDER|REGIONAL HIERARCHY|
      1_GLOBAL_FOLDER|[GLOBAL]|HIERARCHY_CONTEXT|[IS_ROW_PUSHED_DOWN]
      ```
   d. Click **OK**.
12. Now we will create a regional level prompt, do the following:

   a. Drag a ab Text Item **Text Item** into the first column of the table. Enter the text for the prompt in the text item.
   b. Select the table cell and set the location to be ≡ **Horizontally** right and ▤ **Vertically** top.
   c. Make the text bold using the B **Bold** icon.
   d. Drag a ▾ Value Prompt **Value Prompt** object onto your prompt page. This will bring up the Prompt Wizard.

13. On the **Choose Parameter** screen of the Prompt Wizard:
   a. Select the option **Create a new parameter**.
   b. In the box, type a name for the parameter. For example, **Region Entity**.
   c. Click **Next**.
14. On the **Create Filter** screen:
   a. Select the option **Create a parameterized filter**.
   b. In the **Package item** box, click the browse button and navigate to the following query item (the names of these query items are user defined, so they may be different on your system):
      ```
      DEFAULT_REL|GRC_OBJECTS|SOXBUSENTITY_FOLDER|REGIONAL HIERARCHY|
      2_REGIONS_FOLDER|[REGIONS]|ID_FIELDS|[REGIONS_ENTITY_ID]
      ```
   c. In the **Operator** box, use = (the equal sign is the default).
   d. Clear the box for the **Make this filter optional** option.
   e. If you have multiple queries in your report, in the **Apply filter** box, select the query to which the filter should be applied.
   f. Click **Next**.
15. On the **Populate Control** screen:
   a. Select the **Create new query** option.
   b. In the **Name** box, type a name for the query. For example, **Region Entity Prompt**.
   c. In the **Values to display** box, click the browse button and navigate to the following query item:
      ```
      DEFAULT_REL|GRC_OBJECTS|SOXBUSENTITY_FOLDER|REGIONAL HIERARCHY|
      2_REGIONS_FOLDER|[REGIONS]|[REGIONS_NAME00]
      ```
   d. The following values will be displayed in the box:
      ```
      [DEFAULT_REL].[REGIONS].[REGIONS_NAME00]
      ```
   e. In the **Cascade Source** drop down, select **Global Entity**.
   f. Click **Finish**.
16. Now select the prompt you just created and under the **General** category in the Properties pane, change the **Select UI** value to **List Box**.
17. In the Properties pane under **Data**, set the **Sorting** property to be the name of the business entity.
18. If there are a limited number of items in the prompt, you can set the prompt height to a value that shows just the prompts and no white space.
   a. Select the prompt in the table that you want to set the height.
   b. In the Properties pane, under **Positioning**, select **Size and Overflow**.
   c. In the dialog box, set the height to **14 pt** times the number of rows in the prompt.
19. In the **Query Explorer**, select the **Global Entity Prompt** query subject and do the following:

a. Click the **Toolbox** tab on the Insertable Objects pane.

b. Drag and drop a Filter **Filter** object into the Filter pane of the query.

c. In the Expression Definition pane enter the following:

```
[DEFAULT_REL].[REGIONAL].[IS_ROW_PUSHED_DOWN]='N'
```

The data item can be found in the following folder:

```
DEFAULT_REL|GRC_OBJECTS|SOXBUSENTITY_FOLDER|REGIONAL HIERARCHY|
2_REGIONS_FOLDER|[GLOBAL]|HIERARCHY_CONTEXT|[IS_ROW_PUSHED_DOWN]
```

d. Click **OK**.

20. Now we need to add a reprompt button to the prompt page. Do the following:

a. Drag a Prompt Button **Prompt Button** into the table cell below the value prompt you just created.

b. In the Properties pane under **General**, set the **Type** property to **Reprompt**.

c. Drag a Text Item **Text Item** on top of the prompt button. Enter the following text: `Update Regional Entity`.

d. In the Proprieties pane under **Color & Background**, set the **Background Color** property to **Silver**.

e. Select the table cell that the button has been placed in.

f. In the Properties pane under **Positioning**, set the **Size & Overflow** property to **40 px**.

21. Now repeat steps 11-17 and create the Department prompt. You do not need to add a reprompt button after this prompt.

Results

When the report first renders, only the global prompt is populated.

When you select a value from the list and click **Update Regional Entity** the region entity list box is populated with only entities that are direct children of the **Global Financial Services** entity.

Once again, when you select **North American** and click **Update Department Entity**, the **Department Entity** list box is populated with entities that are direct children of the **North America** entity.

Quick Reference for Displaying and Filtering Data

Table 45 on page 128 contains reference information that you can use when creating prompts.

To use this quick reference information, select one of the report data display options, and then implement each of the configurations in that row.

For each data display option, select the following entity values in the **Value prompt** fields:

- For **Package item**, select GEN_ENTITY_ID
- For **Operator**, select the equal sign (=)
- For **Value to use**, select PEN_ENTITY_ID
- For **Value to display**, select PEN_FULL_PATH

Table 45. Entities Quick Reference

Report data display options	Query item value (Report page)	Query item and value in a filter expression (Report query subject)
Only the selected entity This report can support a multi-select pick list prompt. Use "in" for the **Value prompt** operator.	GEN_NAME00	G2P_DISTANCE=0
The selected entity AND the next level below	PEN_NAME00	Not Applicable
The selected entity AND all levels below	CEN_NAME00	Not Applicable
The next level BELOW the selected entity	PEN_NAME00	G2P_DISTANCE=1 Selecting the bottom-level entity in the prompt, results in a null report.
All levels BELOW the selected entity	CEN_NAME00	G2P_DISTANCE=1 Selecting the bottom-level entity in the prompt, results in a null report.
A ROLLUP on the selected entity only Use this option only with a crosstab report.	GEN_NAME00	Not Applicable

Filtering Dimensional Business Entities

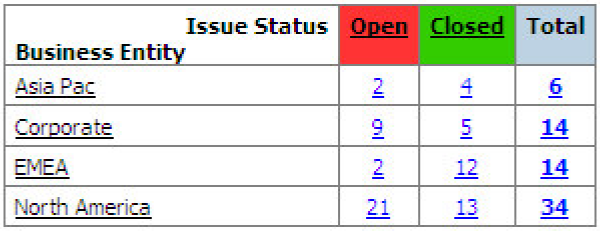
The following topics describe the methodology for filtering entities when using the IBM OpenPages GRC on Cloud dimensional model.

- "Dimensional Entity Context Fields" on page 129
- "The Business Entity Dimensional Model" on page 129
- "Dimensional Prompts" on page 131
- "Using Business Entities for Drill-down/Drill-up" on page 131

To filter entities in the dimensional model, a different set of filter values have been created. These filter values enable you to understand the relationship of the member item to where you are in the hierarchy.

The key concept that you need to understand is that of "push down". The dimensional model is designed so that each time you drill down to another level, the data you see adds up to the data from the previous level.

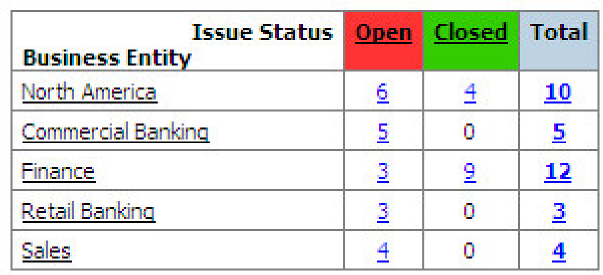
A data item that has been "pushed down" corresponds to data located at the previous level in the hierarchy. For instance if we look at the data in the crosstab shown in Figure 45, we see that there are 21 open issues in the North America business entity and 13 closed issues.



Issue Status Business Entity	Open	Closed	Total
Asia Pac	2	4	6
Corporate	9	5	14
EMEA	2	12	14
North America	21	13	34

Figure 45. Entity Issue Data

When we drill down into North America to further refine our information, we see the following, as shown in Figure 46.



Issue Status Business Entity	Open	Closed	Total
North America	6	4	10
Commercial Banking	5	0	5
Finance	3	9	12
Retail Banking	3	0	3
Sales	4	0	4

Figure 46. Entity Issue Data Showing Push Down

The "pushed down" items are issues that are attached to the North America business entity either directly or through another object. If they were not shown in the crosstab, the data that you drilled into would not add up to what was at the previous level and your users would be curious about the discrepancy.

Dimensional Entity Context Fields

Table 46 is an explanation of the context fields used to filter entities in the dimensional model.

Table 46. Entity Context Fields

Field Name	Description
OP_RECURSIVE_NATIVE_LEVEL	This is a numeric value that represents the native level of the object in the business entity tree.
HIERARCHY_LEVEL	This indicates the level of the hierarchy that you have queried.
IS_ROW_PUSHED_DOWN	This indicates whether this row has been pushed down to this level. Values are "Y" or "N".
ROW_HIERARCHY_LEVEL	This is a numeric value that represents the level of the object as defined in the business entity hierarchy.
ROOT_NATIVE_LEVEL	This is native level of the root of the hierarchy.
ROOT_FULL_PATH	This is the full path to the root of this hierarchy.

All of the entity context fields can be accessed as follows in Report Studio:

```
roleValue(<Field Name>, <Entity Hierarchy>)
```

Where <Field Name> is a quoted string from the above table and the <Entity Hierarchy> is the fully qualified entity hierarchy name. The following is an example of the correct syntax for this function:

```
roleValue('IS_ROW_PUSHED_DOWN',[DEFAULT_DIM].[ENTITY].[ENTITY])
```

The Business Entity Dimensional Model

To better explain the dimensional business entity model, let's look at a sample business entity hierarchy, shown in Figure 47 on page 130:

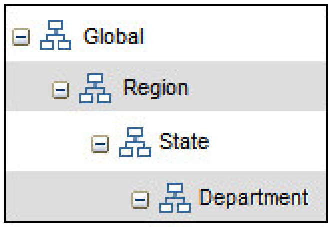
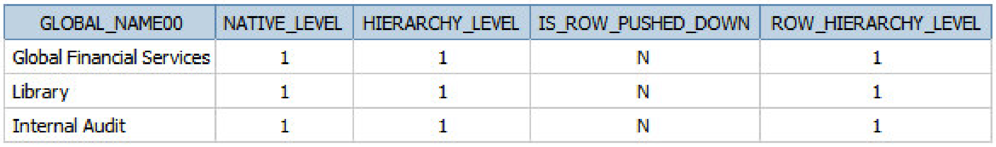


Figure 47. Business Entity Hierarchy

Now if we look at the value of the context fields as a function of the entity level we would see the following, as shown in Figure 48:



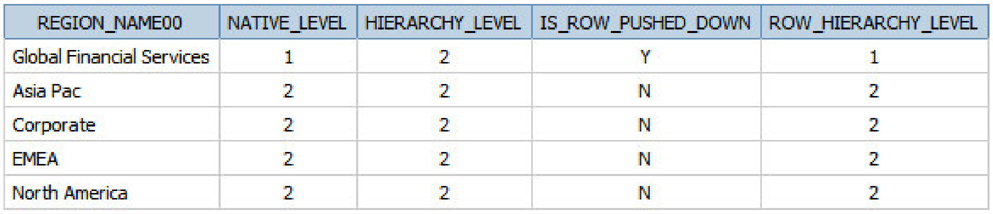
GLOBAL_NAME00	NATIVE_LEVEL	HIERARCHY_LEVEL	IS_ROW_PUSHED_DOWN	ROW_HIERARCHY_LEVEL
Global Financial Services	1	1	N	1
Library	1	1	N	1
Internal Audit	1	1	N	1

Figure 48. Global Level

What this shows is that when we use the global query subject, of the business entity hierarchy, the context variables are returned as follows:

- The [NATIVE_LEVEL] of the entities in the hierarchy is 1 because we created our hierarchy starting at this level.
- The [HIERARCHY_LEVEL] is set to 1 because all objects returned are at the global level of the hierarchy.
- The [IS_ROW_PUSHED_DOWN] values are set to N because at this level there can be no pushed down entities at the top of the hierarchy.
- The [ROW_HIERARCHY_LEVEL] is equal to the [NATIVE_LEVEL] because we created our entity hierarchy starting from the top level.

When we query the region level of the hierarchy, we see the following, as shown in Figure 49:



REGION_NAME00	NATIVE_LEVEL	HIERARCHY_LEVEL	IS_ROW_PUSHED_DOWN	ROW_HIERARCHY_LEVEL
Global Financial Services	1	2	Y	1
Asia Pac	2	2	N	2
Corporate	2	2	N	2
EMEA	2	2	N	2
North America	2	2	N	2

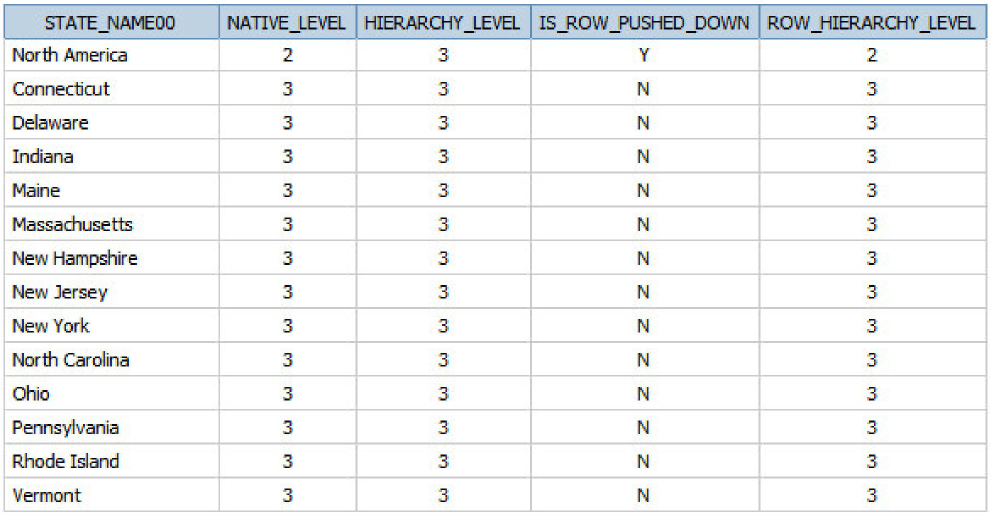
Figure 49. Region Level

What this shows is that when we use the region query subject, of the business entity hierarchy, the context variables are returned as follows:

- The [NATIVE_LEVEL] of the entities is both 1 and 2 due to the Global Financial Services entity that got "Pushed Down" to this level.
- The [HIERARCHY_LEVEL] is set to 2 because all objects returned are at the region level of the hierarchy.

- The [IS_ROW_PUSHED_DOWN] field shows the top level entity Global Financial Services has been "Pushed down" to this level.
- The [ROW_HIERARCHY_LEVEL] is equal to the [NATIVE_LEVEL] because we created our entity hierarchy starting from the top level.

When we query the state level of the hierarchy, we see the following, as shown in Figure 50:



STATE_NAME00	NATIVE_LEVEL	HIERARCHY_LEVEL	IS_ROW_PUSHED_DOWN	ROW_HIERARCHY_LEVEL
North America	2	3	Y	2
Connecticut	3	3	N	3
Delaware	3	3	N	3
Indiana	3	3	N	3
Maine	3	3	N	3
Massachusetts	3	3	N	3
New Hampshire	3	3	N	3
New Jersey	3	3	N	3
New York	3	3	N	3
North Carolina	3	3	N	3
Ohio	3	3	N	3
Pennsylvania	3	3	N	3
Rhode Island	3	3	N	3
Vermont	3	3	N	3

Figure 50. State Level

What this shows is that when we use the state query subject, of the business entity hierarchy, the context variables are returned as follows:

- The [NATIVE_LEVEL] of the entities is 2 or 3 due to the entities that got "Pushed Down" to this level.
- The [HIERARCHY_LEVEL] is set to 3 because all objects returned are at the state level of the hierarchy.
- The [IS_ROW_PUSHED_DOWN] field shows the entities above this level have been "Pushed down" to this level.
- The [ROW_HIERARCHY_LEVEL] is equal to the [NATIVE_LEVEL] because we created our entity hierarchy starting from the top level.

Dimensional Prompts

These are described in “Adding Prompts” on page 95.

Using Business Entities for Drill-down/Drill-up

In order to create a report with drill-up/drill-down you must have already done the following:

- Defined recursive object levels.
- Defined an object type dimension using the recursive object levels.

The instructions for the above can be found in the *IBM OpenPages GRC on Cloud Administrator's Guide*.

Drill-down/drill-up can only be used with the IBM OpenPages GRC on Cloud dimensional model. In order to demonstrate this, we will create a simple chart using an entity hierarchy and the control object.

Do the following:

Procedure

1. Create a chart report.
   a. Select the **Column** chart grouping.
   b. Select the **Column** chart type.
   c. Click **OK**.
2. Drag and drop the following query items into the various chart sections (the names of the fields might be different on your system):
   - Categories (x-axis)

     `DEFAULT|DEFAULT_DIM|OBJECT_TYPE_DIMENSIONS| ENTITY| ENTITY| Global`
   - Default Measures (y-axis)

     `DEFAULT|DEFAULT_DIM|GRC_OBJECTS|SOXCONTROL_FOLDER| SOXCONTROL_FACTS| SOXCONTROL (COUNT_DISTINCT)`
   - Series

     `DEFAULT|DEFAULT_DIM|GRC_OBJECTS|SOXCONTROL_FOLDER| CN_OPERATING_EFFECTIVENESS| CN_OPERATING_EFFECTIVENES`
3. From the **Query Explorer**, select the query.
   a. In the Properties pane under **Miscellaneous**, set the name to **chartMain**.
   b. Drag and drop a **Filter** object into the filter pane.
   c. In the Report Expression pane enter the following:

      -
        ```
        [DEFAULT_DIM].[ENTITY].[ENTITY].[GLOBAL].
                [IS_ROW_PUSHED_DOWN] = 'N'
        ```
   d. Click **OK**.
   e. Repeat steps b-d for the remaining levels of the business entity hierarchy.
4. From the **Page Explorer**, select the main report page.
5. Double-click on the title and set the value to **Control Operating Effectiveness**.
6. Open the **Axis Titles** and do the following:
   a. Select the x-axis title and in the Properties pane.
   b. Under **General** set the **Default Title** property to **No**.
   c. Double-click on the axis title and set the value to **Business Entity**.
   d. Repeat steps a-c for the y-axis and set the value to **Control Count**.
7. Select the **(Default Legend Title)** and do the following:
   a. In the Properties pane under **General** set the **Default Title** property to **No**.
   b. Double-click on the legend title and set the value to **Operating Effectiveness**.
8. In the **Data | Drill Behavior** menu, do the following:
   a. Select the **Allow drill-up and drill-down** check box.
   b. Click **OK**.
9. Run the report.

Results

When you click on Global Financial Services in the report, the report drills down into that entity.

The result shows data associated with the children of Global Financial Services.

Chapter 10. Working with Shared Objects

Shared objects are objects that can be a child of multiple parent object types. For example, multiple parent risk objects can reference a single shared control object or a control object can have multiple parents.

Aggregating Data for Shared Objects

If a shared child object has multiple parents of the same object type, the shared object may be counted multiple times resulting in an inaccurate count.

For example, let’s say a loss of $1000 occurred and the loss was shared between two different loss events for the same identified risk. Figure 51 shows the associations that would exist among Risk, Loss Event and Loss Impact objects:

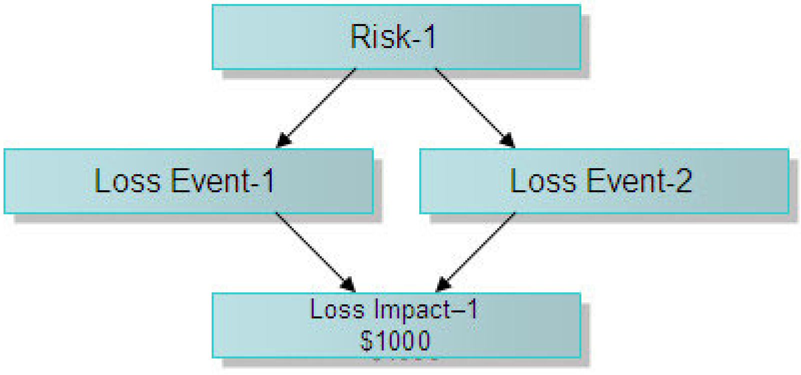


Figure 51. Loss Event Parent-Child Object Associations

If you try to aggregate values for Loss Impact at the Risk level using the following function:

The return value will be 2000.00 (instead of 1000.00) as Loss Impact-1 will be counted twice:

Risk-1 > Loss Event-1 > Loss Impact-1

Risk-1 > Loss Event-2 > Loss Impact-1

To obtain a distinct count of shared objects, use the following formula instead:

```
[average(Loss Impacts for Risk)] * [count (distinct Loss Impacts for Risk)]
```

the return value will be 1000.00 as Loss Impact-1 will be counted only once.

Alternately, we can express the formula mathematically as:

```
average * distinct_count = total/count * distinct_count =
      total * distinct_count/count
```

So, if we are trying to total the Loss Impacts for a Risk as shown in the previous example, we would be performing a total on the repeated Impact and then multiplying by 1/2 to factor out the repetition.

Creating an Aggregation Report For Shared Objects

This procedure creates a sample report that counts the number of unique "Ineffective" and "Effective" controls under a business entity.

Figure 52 shows the association between Entity, Process, Risk, and Control objects in this scenario. Because child Control objects can have multiple parent Risk objects, it is important that the value of a control be counted only once in the report.

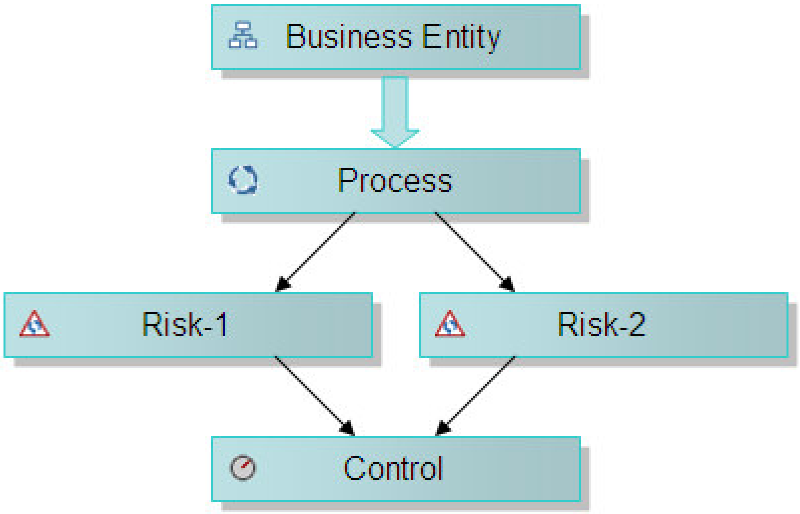


Figure 52. Control Parent-Child Associations

Procedure

1. Create a new list report.
2. Create a child query:
   a. From the **Query Explorer**, open the query.
   b. In the Properties pane under **Miscellaneous**, set the name to **listMain**.
   c. Return to the **Query Explorer** and select the top level folder **Queries**.
   d. Click the **Toolbox** tab.
   e. Drag a **Query** object next to the **listMain** query.
   f. Select the new query object.
   g. In the Properties pane under **Miscellaneous**, set the **Name** to **listMain1**.
3. Add another child query object to the query object you created in the previous step.
   a. Drag a **Query** object onto the **listMain1** query.
   b. Select the new query object.
   c. In the Properties pane under **Miscellaneous**, set the **Name** to **listMain2**.
4. Now you will add query items to the **listMain2** query you created in the previous step.
   a. Select the **listMain2** query to open it.
   b. Click the **Source** tab in the Insertable Objects pane.
   c. Drag the following query items into the **Data Items** box:
      - `DEFAULT|DEFAULT_REL|GRC_OBJECTS|SOXBUSENTITY_FOLDER|[SOXBUSENTITY_GPC]| [SOXBUSENTITY_GPC]|[PEN_NAME00]`

- DEFAULT|DEFAULT_REL|GRC_OBJECTS|SOXBUSENTITY_FOLDER|[SOXBUSENTITY_GPC]| [SOXBUSENTITY_GPC]|ID_FIELDS|[PEN_ENTITY_ID]
- DEFAULT|DEFAULT_REL|GRC_OBJECTS|SOXBUSENTITY_FOLDER|[SOXBUSENTITY_GPC]| [SOXBUSENTITY_GPC]|ID_FIELDS|[GEN_REPORTING_PERIOD_ID]
- DEFAULT|DEFAULT_REL|GRC_OBJECTS|SOXCONTROL_FOLDER|[SOXCONTROL]|ID_FIELDS| [CN_CONTROL_ID]
- DEFAULT|DEFAULT_REL|GRC_OBJECTS|SOXCONTROL_FOLDER|[SOXCONTROL] |ENUMERATION_FIELDS| OPERATING_EFFECTIVENES (ENUMERATION)|[CN_OPERATING_EFFECTIVENES_ID]

5. Add filters to the report:
   a. Select the [GEN_REPORTING_PERIOD_ID] data item you just added and drag it to the Detail Filters pane.
   b. In the Expression Definition pane, define the filter expression:
   ```
   [GEN_REPORTING_PERIOD_ID] = -1
   ```
   c. Click **OK**.
   d. Select the [CN_CONTROL_ID] data item you just added and drag it to the Detail Filters pane.
   e. In the Expression Definition pane, define the filter expression:
   ```
   [CN_CONTROL_ID] is not null
   ```
   f. ☑ Validate the expression and then click **OK**.
6. Now add a data item to the query object:
   a. Click the **Toolbox** tab.
   b. Drag a **Data Item** object into the **Data Items** box of the query.
   c. In the Expression Definition pane of the data item, enter the following expression:
   ```
   IF([CN_OPERATING_EFFECTIVENES_ID] =
      #$SOXCONTROL_OPERATING_EFFECTIVENES_DEFINITION_MAP
         {"Ineffective"}#)THEN ([CN_CONTROL_ID])ELSE (NULL)
   ```
   d. ☑ Validate the expression and then click **OK**.
7. In the Properties pane of the data item:
   a. Change the **Name** property of the data item to **Ineffective Control**.
   b. Change the **Aggregate Function** property to **None**.
8. View the tabular data using the menu item **Run | View Tabular Data**.
9. Open the **listMain1** query subject in the **Query Explorer** and count the number of distinct controls for each entity as follows:
   a. Click the **Source** tab and drag and drop the [PEN_NAME00] query item into the Data Items pane.
   b. In the Properties pane under **Data Item**, set the **Aggregate Function** to **None**.
   c. Click the **Toolbox** tab and drag a **Data Item** object into the Data Items pane.
   d. In the **Expression Definition** box, enter the following expression to count the total number of distinct controls:
   ```
   count(distinct [listMain2].[CN_CONTROL_ID] for [listMain2].[PEN_NAME00])
   ```
   e. ☑ Validate the expression and then click **OK**.
10. In the Properties pane for the data item do the following:
    a. Under **Data Item** set the **Name** property to **Count of Controls**.
    b. Set the **Aggregate Function** property to **None**.

    c. Set the **Rollup Aggregate Function** property to **None**.

11. Count the number of ineffective controls for each entity as follows:
    a. Click the **Toolbox** tab and drag a **Data Item** object into the Data Items pane.
    b. In the **Expression Definition** box, enter the following expression to count the total number of distinct controls:

       ```
       count(distinct [listMain2].[Ineffective Control] for
        [listMain2].[PEN_NAME00])
       ```

    c. ☑ Validate the expression and then click **OK**.
12. In the Properties pane for the data item do the following:
    a. Under **Data Item** set the **Name** property to **Count of Ineffective Controls**.
    b. Set the **Aggregate Function** property to **None**.
    c. Set the **Rollup Aggregate Function** property to **None**.
13. Run the tabular model at this level.
14. Open the **listMain** query subject in the **Query Explorer** and do the following:
    a. Drag and drop the following query items into the Data Items pane:
       - `[PEN_NAME00]`
       - `[Count of Controls]`
       - `[Count of Ineffective Controls]`
    b. In the Properties pane for each of the above controls, set the **Aggregate Function** to **None**.
15. From the **Page Explorer**, navigate to **Page1** and do the following:
    a. Click the **Data Items** tab.
    b. Drag the following objects from the **listMain** query into the list:
       - `[PEN_NAME00]`
       - `[Count of Controls]`
       - `[Count of Ineffective Controls]`
16. Add a business entity prompt as described in “Adding a Business Entity Prompt” on page 56.
17. Run the report.

Results

In this example we had 4 risks that shared 2 controls. If we hadn’t used this method to compute the count of controls, the value would have returned incorrectly as 4.

Counting Shared Objects in Chart Reports

You will need to perform some additional tasks when counting the number of objects under an entity in a chart when some of the objects are shared. For example, reporting on the total controls under a business entity when some controls have multiple parent Risks.

Procedure

1. Create a new Chart report.
    a. Select the **Column** Chart grouping.
    b. Select the **Column** chart type.
    c. Click **OK**.

2. Create a prompt page following the instructions from "Creating a Prompt Page" on page 54
3. Add a business entity prompt as described in "Adding a Business Entity Prompt" on page 56.
4. In the Query Explorer select **Query1** and do the following:
   a. In the Properties pane under **Miscellaneous**, set the **Name** to **graphMain**.
   b. Drag the following query items into the Data Items pane:
      - `DEFAULT|DEFAULT_REL|GRC_OBJECTS|SOXBUSENTITY_FOLDER|[SOXBUSENTITY_GPC]|[CEN_NAME00]`
      - `DEFAULT|DEFAULT_REL|GRC_OBJECTS|SOXBUSENTITY_FOLDER|[SOXBUSENTITY_GPC]|ID_FIELDS|[CEN_ENTITY_ID]`
      - `DEFAULT|DEFAULT_REL|GRC_OBJECTS|SOXCONTROL_FOLDER|[SOXCONTROL]| ID_FIELDS|[CN_CONTROL_ID]`
5. Drag and drop the following query items into the various chart sections:
   - `Categories (x-axis) - [CEN_NAME00]`
   - `Measures (y-axis) - [CN_CONTROL_ID]`
6. In the query explorer, select the query graphMain and do the following:
   a. Select `CN_CONTROL_ID.`
   b. In the Properties pane under the Data Item category, set the **Aggregate Function** property to **Count Distinct**.
   c. Set the **Rollup Aggregate Function** property to **Automatic**.
7. Double-click on the title and set the value to **Control Count**.
8. Open the **Axis Titles** and do the following:
   a. Select the x-axis title and in the Properties pane.
   b. Under **General** set the **Default Title** property to **No**.
   c. Double-click on the axis title and set the value to **Business Entity**.
   d. Repeat steps a-c for the y-axis and set the value to **Control Count**.
9. Select the entire chart and do the following:
   a. In the Properties pane under the **Chart Annotations** category, set the property **Legend** to **Hide**.
10. Run the report.

Results

The report displays the results of the control count which is filtered by the business entity that was selected. You should choose a business entity where you can verify the values in the chart.

Listing Shared Objects in List Reports

There is a property called **Auto Group & Summarize** attached to all query subjects that helps eliminate or show duplicate records. The property can be set as follows:

- **Yes** (this is the default) if you want to display each control only once no matter how many parent risks it has.
- **No** if you want to display a control once for each risk object to which it is associated. So, if a control has three parent risks, then the control will be displayed three times.

Depending on how you have written your report and whether showing the shared objects is your objective, you will need to adjust this setting on your query subject.

Chapter 11. Working With Recursive Objects

To see all items attached to the different levels of the recursive hierarchy in your report, there is a construct called a reporting triangle.

Reporting Triangles

In Chapter 9, “Working with Business Entities,” on page 119, we discussed working with recursive business entity objects. There are however three other recursive objects in IBM OpenPages GRC on Cloud that will exist in your system if you have the appropriate modules:

- Sub-Process
- Sub-Mandate
- Sub-Account

Typically, from a report author’s point of view, you want to see all items attached to the different levels of the recursive hierarchy in your report. In order to do this, you would have to create a relationship in your namespace where an object had more than one parent, but this is not allowed.

In order to work around this problem, we have created a construct called a reporting triangle. There are three types of triangles that can be created, as shown in the following figures:

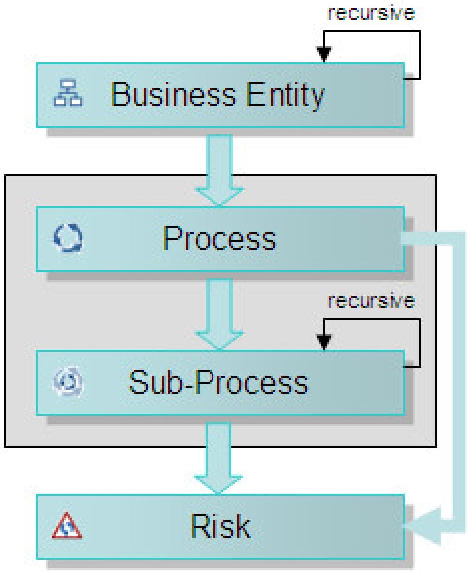


Figure 53. Process, Sub-Process, Risk Triangle

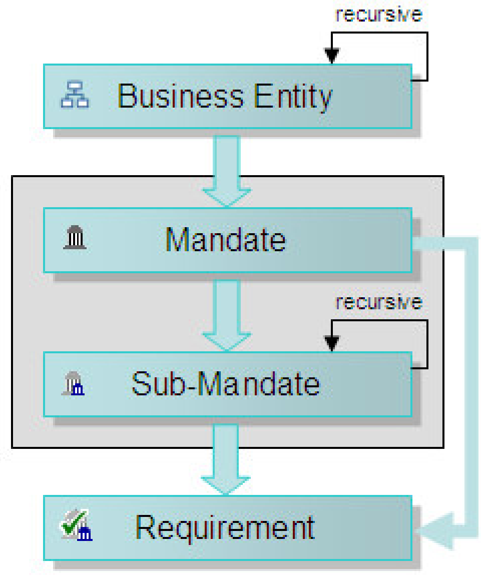


Figure 54. Mandate, Sub-Mandate, Requirement Triangle

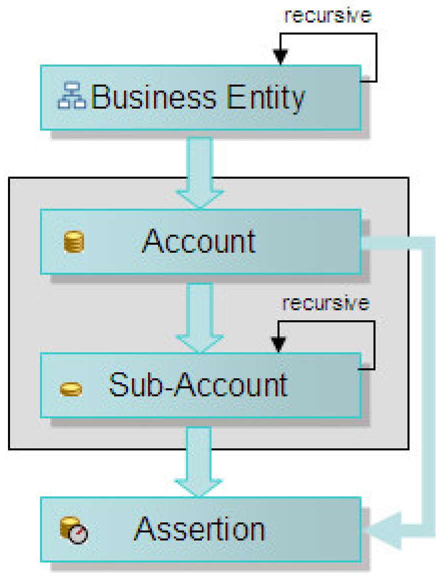


Figure 55. Account, Sub-Account, Assertion Triangle

The common factor here is that the recursive object is the center of the triangle. Thus by creating a triangle whenever you deal with a recursive object, the report author does not have to write separate code to handle the case where the data is attached to the main object and the case where the data is attached to the sub-object.

Now if the triangle is not supported in the namespace you are using, you can always union the results of two namespace to yield the same values as if the triangle existed.

The *IBM OpenPages GRC on Cloud Administrator's Guide* describes how to create a triangle. The only caveat is that each time you create a new triangle or add one to another namespace you must run a SQL script that enables the specific triangle relationship and then you must recreate the schema.

Chapter 12. Internationalization and Localization

The IBM OpenPages GRC on Cloud application is fully internationalized and localized.

Data Translation

IBM OpenPages GRC on Cloud contains static and application (instance) data.

Generally, all static data is translated either through the reporting framework or by the report while application data is not translated, with the exception of enumerated strings.

The language in which a user views this data is based on the user's locale setting which is set in OpenPages GRC on Cloud when the user is created or later modified. This is what determines how the report will be translated.

It is possible to set the user's locale in Cognos Connection, which would separate it from the locale set in the OpenPages GRC on Cloud application. This is typically done by the report author to leave a particular user with one locale in the OpenPages GRC on Cloud application and another when using reports. See "Setting Your Locale in Cognos Connection" on page 44.

Static Data

There are four categories of static data: dynamic labels, fixed text fields, button text, and text in prompts.

- Dynamic label data - This includes the displayed label text for object types (such as business entity, account, control, etc.) and object field names (such as Operating Effectiveness, etc.) as well as buttons, labels, text boxes, column headings, etc. This data (and the translations) are created when fields are defined in the IBM OpenPages GRC on Cloud application.
- Fixed text field data - This includes any fixed text string that has been added to the report by the report author using a text object.
- Button Text - This includes the text displayed by any prompt button or the buttons on the prompt page.
- Static Text in Prompts - This includes any static text added to a prompt (such as All) or a prompt that is all static text.

Translating Dynamic Label Data

When you create a report using the Report Design Language, both the Insertable Objects pane and the report design window show the object and field names using the internal system name.

When you execute the report, the static text is displayed in the user's locale defined in IBM OpenPages GRC on Cloud:

- If the user's locale is set to Report Design Language (RDL), the titles of the columns are in RDL, but the data is in English.
- If the user's locale is set to US English in the OpenPages GRC on Cloud application, when the report is executed, the column titles are translated to US English.

- If the user’s locale is set to Spanish in the OpenPages GRC on Cloud application, when the report is executed in the Spanish Locale, the column titles are translated to Spanish.

These examples demonstrate why we use the Report Design Language as the default language for designing the report and how the system internally translates the static titles based upon definitions loaded in the OpenPages GRC on Cloud application. This saves time and effort by the report author if reports are being created for a multi-locale company.

User-entered application data is not translated. If the data had been entered by a Spanish user, you would see Spanish descriptions for every locale because application data is not translated by locale.

Translating Fixed Text Field Data

Any text objects (report titles, subtitles, and labels) that you add to a report will need to be localized. You can use condition variables to perform this translation on the static text. In order to do this, a condition variable must first be created.

The following describes the steps for creating and using a condition variable.

Procedure

1. Open the **Condition Explorer** menu and then select **Variables**.
2. Drag and drop a **Report Language Variable** object from the toolbox into the **Variables** box.
3. After you drop the **Report Language Variable** into the **Variables** list box, the Languages dialog box will appear. Do the following:
   a. Select the languages for which the text needs to be localized. The current list of supported OpenPages GRC on Cloud languages are as follows:
      - English (United Kingdom)
      - English (United States)
      - French (France)
      - German (Germany)
      - Italian (Italy)
      - Japanese (Japan)
      - Spanish

   b. Click **OK**.
4. In the **Properties** pane:
   a. Enter the name of the variable in the **Name field**. Use **Localization** as the name.

   To convert a text field to a specific language in your report, do the following:
5. Select one of the labels that needs to be translated. In this example, the report title **Test List** was selected.
6. In the Properties pane, of the selected text item, under the **Conditional** group:
   a. In the **Text Source Variable** property, select **Localization** variable you just created.
7. In the **Condition Explorer**, select one of the languages that is present under the language variable.
8. In the Properties pane under **Text Source**:
   a. In the **Text** property enter the localized string of the label. For example, **Prueba Lista**.

   b. Click **OK**.
   c. Repeat steps 3-4 and define the text string for all languages.
   d. When the languages strings are all defined, select **No Variable** in the **Condition Explorer**.
9. View the report:
   a. Login using a user who has their locale set to Spanish (if you are not already logged with a Spanish locale user).
   b. Run the report.

Results

The title of the report now appears translated.

The above methodology can be followed for any text label in your report. The condition variable that was created can be re-used for each object that needs translation.

Translating Button Text

Typically text buttons on the prompt page do not require localization because it is handled by the reporting tool.

On the other hand, when you create your own prompt button and need to localize its text, follow this procedure.

Procedure

1. Identify the button that you wish to localize.
2. Select a **Text Item** from the toolbox and do the following:
   a. Drop the **Text Item** on top of the button.
   b. Enter **Update Line of Business** as the default text string.
   c. In the properties pane of the **Text Item**, enter the condition variable **Localization** as the **Text Source Variable** property.
3. In the **Condition Explorer**, do the following:
   a. While the **Text Item** is selected, select each language in the **Localization** variable that you wish to specify.
   b. Enter the appropriate text string in the **Text** property.
   c. Repeat for each language you wish to specify.
4. View the report:
   a. Login using a user whose locale is different than the current user.
   b. Run the report.

Translating Static Text in Prompts

Static text in prompts is another area of reporting that needs to be translated to the user's locale. Follow this procedure to localize the static text in a prompt.

Procedure

1. Identify the prompt that you wish to localize static text within. In this example we will be localizing the static text in the **Region** prompt.
2. Select the **Region** prompt and do the following:
   a. Select the **Static Choices** property in the Properties pane.
   b. In the **Variables** drop down select **Localization,** the condition variable that was created for translating text to other languages.

c. In the **Value** drop down select **English (United States)**.

d. Using the add icon, enter the values for the static selection list for English.

e. Now select **Spanish** in the **Value** drop down and enter the Spanish values using the edit icon.

f. Click **OK** to save the values.

3. View the report:

a. Login using a user who has a default locale set to **Spanish** (if you are not already logged with a Spanish locale user).

b. Run the report.

Results

The text in the prompt appears translated in Spanish.

When your locale is set to **English (United States)** the Region prompt appears with its English values.

Application Data

There are four categories of application data: user entered data, enumerated strings, currency fields, and date fields.

- User entered data - This includes all the data entered by users. This is not translated.
- Enumerated string data - This includes all data selected by users from a list of enumerated string values. This data (and the translations) are created when the list of enumerated strings are defined in the IBM OpenPages GRC on Cloud application.
- Currency fields - This includes any data that is configured as a currency.
- Date fields - This includes any data that is formatted as a date or date/time.

Translating Single Valued Enumerated Strings

A single valued enumerated string field allows a user to select only one value from the list. For example, the risk object type has an enumerated string field called inherent impact from which only one value may be selected from the list.

In a single valued enumerated string field, the string is translated inside the query subject based on the user's locale setting.

When you are logged into the application with the user's locale set to Spanish, the same field appears.

When the inherent impact enumerated string was defined through the IBM OpenPages GRC on Cloud application, the Spanish values for the string were also entered in the system.

Framework Folder Structure for Single-Value Enumerations:

Each query subject for an object contains a folder called ENUMERATION_FIELDS with the enumerated strings for that particular object.

The example in Figure 56 on page 149 shows the SOXRisk object and the INHERENT_IMPACT enumeration that is single valued.

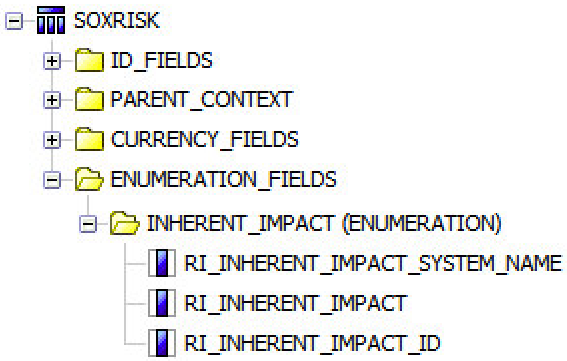


Figure 56. SOXRisk Folder Structure

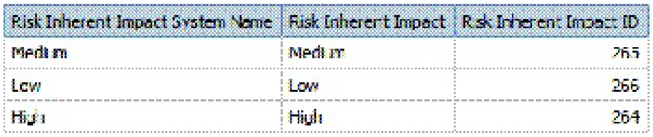
Field Definitions for Single-Value Enumerations:

Each single valued enumeration has the following query items: `<field_name>_SYSTEM_NAME`, `<field_name>`, and `<field_name>_ID`.

- `<field_name>_SYSTEM_NAME` - Contains the string values in Report Design Language.
- `<field_name>` - Contains the string values based on the logged on user's IBM OpenPages GRC on Cloud application locale setting.
- `<field_name>_ID` - contains a numeric identifier for an enumerated string value. This identifier is not guaranteed to be the same across installations.

When displaying the value in a report `<field_name>` should always be used. When filtering on this value, use the `<field_name>_ID` field because it is indexed.

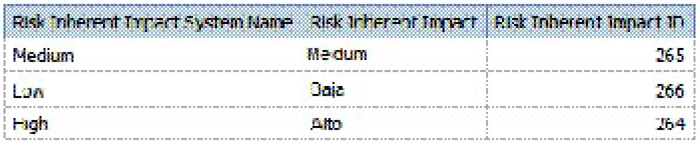
Figure 57 shows the Inherent Impact enumerations in the US English locale. Both the system name and the locale name are the same because no change was made to the English value when the data item was loaded in the system. It is possible that the system name (in RDL) and the English name are different.



Risk Inherent Impact System Name	Risk Inherent Impact	Risk Inherent Impact ID
Medium	Medium	265
Low	Low	266
High	High	264

Figure 57. Inherent Impact in US English

Figure 58 shows the Inherent Impact enumerations in the Spanish locale. The system name remains in RDL while the locale name is translated to Spanish.



Risk Inherent Impact System Name	Risk Inherent Impact	Risk Inherent Impact ID
Medium	Medium	265
Low	Bajo	266
High	Alto	264

Figure 58. Inherent Impact in Spanish

Translating Multivalued Enumerated Strings

A multivalued enumerated string field allows a user to select multiple string values from a list.

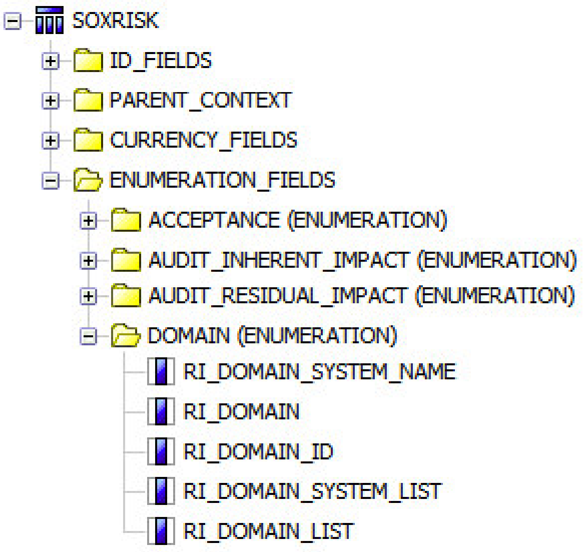
In a multivalued enumerated string field, the string is translated inside the query subject based on the user's locale setting.

When the domain enumerated string was defined through the IBM OpenPages GRC on Cloud application, the Spanish values for the string were entered in the system and thus the users will see the translated values in the Spanish locale.

Framework Folder Structure for Multivalued Enumerations:

The folder structure for enumerations contains both single valued and multivalued enumerations.

Figure 59 show the enumerations folder for the SOXProcess object expanded with the domain folder opened to show its fields. The domain field is multivalued.



- SOXRISK
  - ID_FIELDS
  - PARENT_CONTEXT
  - CURRENCY_FIELDS
  - ENUMERATION_FIELDS
    - ACCEPTANCE (ENUMERATION)
    - AUDIT_INHERENT_IMPACT (ENUMERATION)
    - AUDIT_RESIDUAL_IMPACT (ENUMERATION)
    - DOMAIN (ENUMERATION)
      - RI_DOMAIN_SYSTEM_NAME
      - RI_DOMAIN
      - RI_DOMAIN_ID
      - RI_DOMAIN_SYSTEM_LIST
      - RI_DOMAIN_LIST

Figure 59. Folder Structure

Field Definitions for Multivalued Enumerations:

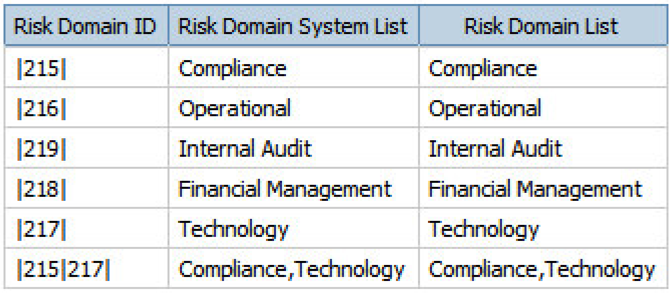
Each multivalued enumeration has the following query items: `<field_name>_SYSTEM_NAME`, `<field_name>`, `<field_name>_ID`, `<field_name>_SYSTEM_LIST`, and `<field_name>_LIST`.

- `<field_name>_SYSTEM_NAME` - Contains the string values in Report Design Language. There will be one record for each enumeration selected. This returns one record for each value selected by the user.
- `<field_name>` - Contains the translated string values based on the user's IBM OpenPages GRC on Cloud application locale setting. This returns one record for each value selected by the user.
- `<field_name>_ID` - Contains a numeric identifier for an enumerated string value. This identifier is not guaranteed to be the same across installations. This is returned as a pipe "|" separated list.

- `<field_name>_SYSTEM_LIST` - Contains a comma separated list of the string values based in Report Design Language.
- `<field_name>_LIST` - Contains a comma separated list of the string values based on the user's locale setting.

In order to obtain a comma separated list of the translated values of the string, the report author needs to select the <field_name>_LIST field and use that in the report. This is the best field to use to display the value of a multivalued enumerated string.

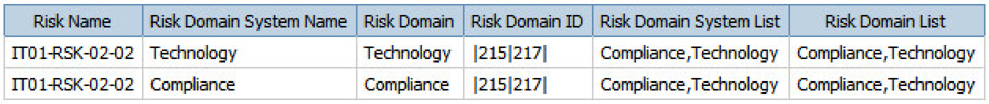
Figure 60 shows the domain fields, in the US English locale, for risks that are loaded in the system. You also only return one record per risk when the any of the three aggregated fields are used.



Risk Domain ID	Risk Domain System List	Risk Domain List
|215|	Compliance	Compliance
|216|	Operational	Operational
|219|	Internal Audit	Internal Audit
|218|	Financial Management	Financial Management
|217|	Technology	Technology
|215|217|	Compliance,Technology	Compliance,Technology

Figure 60. Domain Multivalued Fields in English

When you include RI_DOMAIN_SYSTEM_NAME or RI_DOMAIN fields you will see the following (a filter was placed in the report for a risk that had multiple domain values):



Risk Name	Risk Domain System Name	Risk Domain	Risk Domain ID	Risk Domain System List	Risk Domain List
IT01-RSK-02-02	Technology	Technology	|215|217|	Compliance,Technology	Compliance,Technology
IT01-RSK-02-02	Compliance	Compliance	|215|217|	Compliance,Technology	Compliance,Technology

Figure 61. Domain All Fields in English

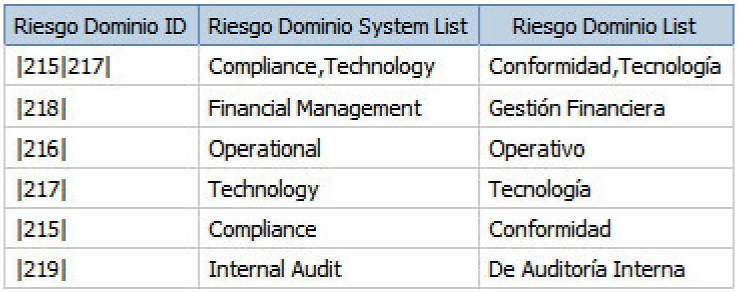
You see that these fields return a single record for each value that the user selected.

These two fields are used for filtering and not displaying data to the user because of the duplicate records. The typical filter used for risk domain would look as follows:

```
[DEFAULT_REL].[SOXRISK].[RI_DOMAIN] in ?Domain? or ?Domain? = 'All'
```

This filter is used in conjunction with a multi-select prompt to filter domain values. It also includes a value for **All** which was added as a static value to the prompt.

Now when we switch to a user with a Spanish locale, we see the following for the aggregated fields of Risk Domain ID, Risk Domain System List and Risk Domain List:

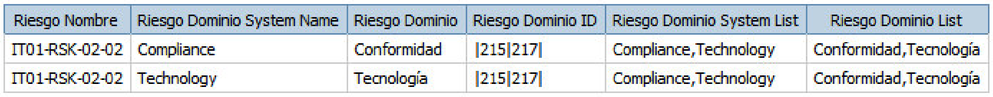


<table>
<tr><th>Riesgo Dominio ID</th><th>Riesgo Dominio System List</th><th>Riesgo Dominio List</th></tr>
<tr><td>|215|217|</td><td>Compliance,Technology</td><td>Conformidad,Tecnología</td></tr>
<tr><td>|218|</td><td>Financial Management</td><td>Gestión Financiera</td></tr>
<tr><td>|216|</td><td>Operational</td><td>Operativo</td></tr>
<tr><td>|217|</td><td>Technology</td><td>Tecnología</td></tr>
<tr><td>|215|</td><td>Compliance</td><td>Conformidad</td></tr>
<tr><td>|219|</td><td>Internal Audit</td><td>De Auditoría Interna</td></tr>
</table>

Figure 62. Domain Multivalued fields in Spanish

We see the Risk Domain ID values and the Risk Domain System List values are identical to the values seen in Figure 60 on page 151. The Risk Domain List and the field header titles are now in the Spanish locale of the user.

The example in Figure 56 on page 149 shows the SOXRisk object and the Inherent Impact enumeration that is single valued. Figure 63 shows what we see when we add the Risk Domain System Name and Risk Domain to the report (a filter was added to the report for a risk with multiple domain values):



<table>
<tr><th>Riesgo Nombre</th><th>Riesgo Dominio System Name</th><th>Riesgo Dominio</th><th>Riesgo Dominio ID</th><th>Riesgo Dominio System List</th><th>Riesgo Dominio List</th></tr>
<tr><td>IT01-RSK-02-02</td><td>Compliance</td><td>Conformidad</td><td>|215|217|</td><td>Compliance,Technology</td><td>Conformidad,Tecnología</td></tr>
<tr><td>IT01-RSK-02-02</td><td>Technology</td><td>Tecnología</td><td>|215|217|</td><td>Compliance,Technology</td><td>Conformidad,Tecnología</td></tr>
</table>

Figure 63. Domain All Fields in Spanish

You once again see that these fields return a single record for each value that the user selected.

Translating Date Fields using Predefined Date Formats

The formats described here can be used, or the report authors can specify their own custom formats.

There are predefined date formats for all OpenPages GRC on Cloud locales.

The following table specifies the format for each locale.

Table 47. Predefined Date Formats

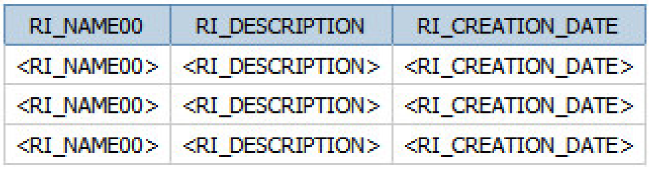
Locale	Formatted Date Format
English (United Kingdom)	dd mmm yyyy
English (United States)	mmm dd, yyyy
French (France)	dd mmm. yy
German (Germany)	dd.mm.yyyy
Italian (Italy)	dd/mmm/yy
Japanese (Japan)	yyyy/mm/dd
Spanish (Spain)	dd/mm/yyyy

Follow this procedure to use a pre-defined date format.

Procedure

1. In this example we are going to localize the risk creation date. Create a simple report displaying the risk name, risk description and risk creation date.

2. Once you have your report set up, it should look as follows:



RI_NAME00	RI_DESCRIPTION	RI_CREATION_DATE
<RI_NAME00>	<RI_DESCRIPTION>	<RI_CREATION_DATE>
<RI_NAME00>	<RI_DESCRIPTION>	<RI_CREATION_DATE>
<RI_NAME00>	<RI_DESCRIPTION>	<RI_CREATION_DATE>

3. Now select the **RI_CREATION_DATE** date field and do the following:
   a. Right mouse click and select **Style | Data Format...**
   b. Set the **Format type** to **Date** but **DO NOT** set any of the properties.
   c. Click **Ok**.
4. Save the report.

Results

By setting the field to be a date with no formatting properties, the date now has the default format for the US English locale as seen in the table at the beginning of this topic.

Now run this report with a user that has a locale of Spanish and the date now has the default format for the Spanish locale as seen in the table at the beginning of this topic.

Translating Date Fields using Custom Date Formats

To create a custom date format for multiple languages, you will have to both localize the date field with a conditional format and set the formats for each language.

Procedure

1. In this example we are going to localize the risk creation date to have custom formats for the German and French locales. Create a simple report displaying the risk name, risk description and risk creation date.
2. Select the **RI_CREATION_DATE** data field and do the following:
   a. In the Properties pane, under the **Conditional** heading, set the **Style Variable** to the conditional variable Localization.
3. In the **Condition Explorer** select **Germany** that is present under the **Localization** variable.
4. Right mouse click and select **Style | Data Format...**
5. The Data Format screen will appear and you will need to set the following properties:
   a. Set the **Format type** to **Date**.
   b. Set the **Date Separator** to . (period).
   c. Set the **Date Ordering** to **Day,Month,Year**.
   d. Set the **Display Years** to **Show century**.
   e. Set the **Display Months** to **Short name**.
   f. Set the **Display Days** to **Two digits**.
6. Click **OK**.
7. In the **Condition Explorer** select **French** that is present under the **Localization** variable.
8. Right mouse click and select **Style | Data Format...**

9. The Data Format screen will appear and you will need to set the following properties:
   a. Set the **Format type** to **Date**.
   b. Set the **Date Separator** to . (period).
   c. Set the **Date Ordering** to **Day,Month,Year**.
   d. Set the **Display Years** to **Show century**.
   e. Set the **Display Months** to **Short name**.
   f. Set the **Display Days** to **Two digits**.
10. Click **OK**.
11. Save the report.

Results

By opting to set custom date field properties, the date now has the format that was defined custom instead of the default date format for the French Locale.

Now run this report with a user that has a locale of German and the date now has the custom format for the German locale instead of the default date format.

Translating Currency Fields using Pre-Defined Formats

The formats described here can be used, or the report authors can specify their own custom formats.

There are pre-defined currency formats for all IBM OpenPages GRC on Cloud locales.

The following table specifies the format for each locale.

Table 48. Pre-Defined Currency Formats

Locale	Formatted Currency Format
English (United Kingdom)	£ NNN,NNN.NN
English (United States)	$ NNN,NNN.NN
French (France)	NNN NNN,NN €
German (Germany)	NNN.NNN,NN €
Italian (Italy)	€ NNN.NNN,NN
Japanese (Japan)	¥ NNN,NNN
Spanish (Spain)	NNN.NNN,NN €

Follow this procedure to use a pre-defined format.

Procedure

1. In this example we are going to localize the entity risk appetite. Create a simple report displaying the entity name, entity description and entity risk appetite all for the grandparent level.
2. Now select the **GEN_RISK_APPETITE_BA** currency field and do the following:
   a. Right mouse click and select **Style | Data Format...**
   b. Set the **Format type** to **Currency** but **DO NOT** set any of the properties.
   c. Click **Ok**.

3. Save the report.

Results

By setting the field to be a currency with no formatting properties, the currency now has the default format for the US English locale as seen in the table at the beginning of this topic.

Now run this report with a user that has a locale of Spanish.

The currency now has the default format for the Spanish locale as seen in the table at the beginning of this topic.

Translating Currency Fields using Custom Formats

To create a custom currency format for multiple languages, you will have to both localize the currency field with a conditional format and set the formats for each language.

Procedure

1. In this example we are going to localize the entity risk appetite. Create a simple report displaying the entity name, entity description and entity risk appetite all for the grandparent level. We will be setting the format for German and UK English.
2. Select the **GEN_RISK_APPETITE_BA** data field and do the following:
   a. In the Properties pane, under the **Conditional** heading, set the **Style Variable** to the conditional variable Localization.
3. In the **Condition Explorer** select **Germany** that is present under the **Localization** variable.
4. Right mouse click and select **Style | Data Format...**
5. The Data Format screen will appear and you will need to set the following properties:
   a. Set the **No. of Decimal Places** to **0**.
   b. Set the **Decimal Separator** to , (comma).
   c. Set the **Use Thousands Separator** to **Yes**.
   d. Set the **Thousands Separator** to , (comma).
6. Click **OK**.
7. In the **Condition Explorer** select **UK English** that is present under the **Localization** variable.
8. Right mouse click and select **Style | Data Format...**
9. The Data Format screen will appear and you will need to set the following properties:
   a. Set the **Currency Symbol Position** to **End**.
10. Click **OK**.
11. Save the report.

Results

By opting to set custom date field properties (separator is a comma), the date now has the format that was defined custom instead of the default date format for the German Locale.

Now run this report with a user that has a locale of UK English.

The currency now has the custom format (currency symbol at the end) for the UK English locale instead of the default currency format.

Using Parameter Maps with Translated Enumerated String Filters

In order for filters based upon enumerated strings to be created that are independent of locales, you must use parameter maps. The parameter map is considered to be the RDL locale so you must use English values and perform your comparison on ID fields.

For each enumerated string definition, a corresponding parameter map is created in the framework. The parameter map returns the ID for each enumerated string value passed to it.

The naming convention for parameter maps is as follows:

```
<Object Name>_<Enumerated String>_DEFINITION_MAP
```

Parameter maps should be used in all filters involving enumerations that can be translated because they will always return the same ID regardless of what locale is set for the user.

For example, we want to filter on a prompt value of High for the inherent impact field on the SOXRisk object. We would type the following into a filter field:

```
?Inherent Impact Id? = #$SOXRISK_INHERENT_IMPACT_DEFINITION_MAP("High")#
```

`?Inherent Impact Id?` is the prompt parameter defined in the report.

Chapter 13. Using Predefined Database Functions

IBM OpenPages GRC on Cloud has exposed certain database functions for use by report authors in order to solve certain technical problems that are common between environments. This chapter describes those functions and gives examples on how to use them.

Passing Parameters

In SQL, parameters are passed positionally (first value to the first parameter, second value to the second parameter, third value to the third parameter, and so forth).

Although you cannot, for example, pass a value to the third parameter while skipping the second parameter, you can, however, take advantage of parameters with default values.

Example

The following predefined function (found “Getting Tree Path Resource IDs” on page 182) has 7 parameters:

```
"OP_RPS_AUX.IS_REL_PARENT_CHILD" (P1, P2, P3, P4, P5, P6, P7)
```

Parameters **P1** and **P2** do not have default values, parameters **P3** - **P7** have default values.

You could, for example, specify values for the **P1** - **P3** parameters and omit values for parameters **P4** - **P7**. In this case, the function would use the default values for **P4** - **P7**, such as in the following sample expression:

```
"OP_RPS_AUX.IS_REL_PARENT_CHILD" ([DEFAULT_REL].[SOXPROCESS].[PR_PROCESS_ID],
  [DEFAULT_REL].[SOXCONTROL].[CN_CONTROL_ID], -1)
```

Because of the positional nature of parameters in SQL, if you wanted to use a value other than the default value for the *P6* parameter (the default is ‘Y’ but we want ‘N’), you would have to include the default values for **P4** (which is ‘N’) and **P5** (which is ‘Y’) as well, such as in the following sample expression:

```
"OP_RPS_AUX.IS_REL_PARENT_CHILD" ([DEFAULT_REL].[SOXPROCESS].[PR_PROCESS_ID],
   [DEFAULT_REL].[SOXCONTROL].[CN_CONTROL_ID], -1, 'N', 'Y', 'Y')
```

Determining an Object Type ID

Some of the predefined database functions have parameters that require an object type ID (a numeric identifier for the object type). To determine a given object type’s numeric ID value, you can either run a query using a SQL tool or create and run a report.

Using SQL to Obtain Object IDs

To obtain a list of objects and object type IDs from the database, execute a SQL command.

Using a SQL tool, execute the following SQL command to obtain a list of objects and object type IDs from the database.

```
SELECT
    name,
    assettypeid
FROM
    assettypes
ORDER BY
    name
```

From one environment to another the asset type IDs may be different.

Some SQL functions are different in DB2 than in Oracle. If you use an Oracle function that is listed in the following table, you must substitute the corresponding DB2 function.

Table 49. Mapping Oracle to DB2 functions

Substitute this DB2 function...	**For this Oracle function...**
INSTR (P1, P2, P3, P4 {CODEUNITS32})	INSTR (P1, P2, P3, P4)
SUBSTR2	SUBSTR
LISTAGG	STRAGG
OP_LENGTH (P1)	LENGTH (P1)

Where P1, P2, P3, P4 represent parameters.

The following example code shows the Oracle INSTR function:

```
INSTR([CEN_FULL_PATH], '/', 1, 2)
```

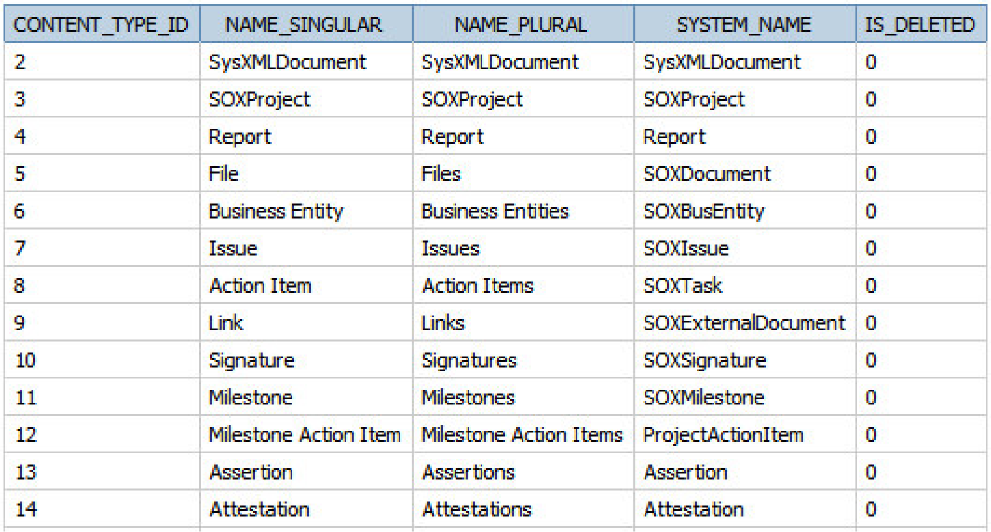
For DB2, you would substitute this code for the INSTR function:

```
INSTR([CEN_FULL_PATH], '/', 1, 2, {CODEUNITS32})
```

Using a Report to Obtain Object IDs

You can create a report and then run it to obtain the object names and object type IDs.

For example, the CONTENT_TYPE_ID column of the sample report, shown in the figure below, lists all the available object type IDs that you could use. Let's say you want to use the OP_RPS_AUX.IS_REL_PARENT_CHILD_PRIM function. The P1 and P3 parameters of this function require the object type IDs of the child and parent object type. You could use, for example, "18" in P1 (child RiskAssessment object type ID) and "4" in P3 (parent SOXBusEntity).



CONTENT_TYPE_ID	NAME_SINGULAR	NAME_PLURAL	SYSTEM_NAME	IS_DELETED
2	SysXMLDocument	SysXMLDocument	SysXMLDocument	0
3	SOXProject	SOXProject	SOXProject	0
4	Report	Report	Report	0
5	File	Files	SOXDocument	0
6	Business Entity	Business Entities	SOXBusEntity	0
7	Issue	Issues	SOXIssue	0
8	Action Item	Action Items	SOXTask	0
9	Link	Links	SOXExternalDocument	0
10	Signature	Signatures	SOXSignature	0
11	Milestone	Milestones	SOXMilestone	0
12	Milestone Action Item	Milestone Action Items	ProjectActionItem	0
13	Assertion	Assertions	Assertion	0
14	Attestation	Attestations	Attestation	0

Figure 64. Sample Report Showing the Object Type Numeric Identifier

Procedure

1. Create new List report.
2. Navigate to: DEFAULT_REL|MISCELLANEOUS
3. Under the: LOCALIZED_CONTENT_TYPES query subject, drag the SYSTEM_NAME and CONTENT_TYPE_ID into the report template.
4. Change the **Sort** field value to **Sort ascending** of the SYSTEM_NAME heading.
5. Run the report in PDF.
6. Save the resulting PDF file to your local drive.

Determining Database Table Names

Database table names have specific naming conventions.

Where:

<OBJECT TYPE> is the name of the type of object.

<ENUMERATION> is the name of a multivalued enumerated string field.

This naming convention...	Indicates this is a...
RT_<OBJECT TYPE> (RT_ with a single underscore)	Main table For example, RT_RISKASSESSMENT
RT_<OBJECT TYPE>_<ENUMERATION> (RT_ with a single underscore)	Multivalued Enumeration table For example, RT_RISKASSESSMENT_DOMAIN
RT__<OBJECT TYPE>_<OBJECT TYPE> (RT__ with a double underscore)	Bridge table between 2 object types For example, RT__RISKASSESSMENT_PROCESS

Important:

- Oracle Database has a limitation of 30 characters for a table name. Some table names will be truncated.

- ‘RT_’ tables must be all uppercase and enclosed in single-quotes, for example 'RT_RISK'.
- Any ‘RT_’ entry with more than one underscore in the string is a multivalued selection table. For example, in Risk records there is a multivalued selection field labeled "Financial Statement Assertions". That field information is stored in the RT_RISK_FINANCIAL_STATEMENT_AS table.

Obtaining a list of RT table names

Using a SQL tool, execute the following command:

```
SELECT
  object_name
FROM
  dba_objects
WHERE
   object_name like 'RT_%'
ORDER BY
  object_name
```

The result is the list of all RT tables in the database.

Determining Database Table Column Names

There are three techniques for determining database table column names.

Many table column names can be determined by one of the following methods:

- Expanding a query subject in Report Studio to obtain the field names
- Opening up the tables folder in a SQL tool and expanding the table definition you are seeking
- Using the DESCRIBE <table_name> function in SQL to obtain the table definition

Note:

- Long field names that exceed 22 characters are truncated after character 22 (for example, FINANCIAL_STATEMENT_ASSERTIONS becomes FINANCIAL_STATEMENT_AS). However, if two or more field names have the same first 22 characters, then the names are truncated at character 20 and a two-digit sequential identifier is appended. For example:

  ```
  FINANCIAL_STATEMENT_ASSERTIONS becomes FINANCIAL_STATEMENT_00
  FINANCIAL_STATEMENT_ASSERTIONS_MINOR becomes FINANCIAL_STATEMENT_01
  ```

- If multiple attributes are needed, you can concatenate multiple field values with a delimiter, for example:

  ```
  'NAME00' || '||' || '''=''' || '||' || 'FULL_PATH'
  ```

  The concatenated and delimited Name plus 'FULL_PATH’ is a performance boost to avoid calling the function twice. You can only use this method with regular, non-enumerated properties.
- The truncation of variables will be different in different environments.

Using General Functions

This section describes general database functions.

Converting Multivalue Enumerations to Names

This function converts a token-separated list of enumerated value IDs into a separated and sorted list of localized names.

Report authors can use this function to turn a multivalue enumeration ID into a single string. This function eliminates the need to use custom SQL statements or additional query subjects.

Important: For Oracle, there is a maximum limit of 4000 characters for a single string.

Syntax

This function has the following syntax:

```
"OP_RPS_AUX.CONVERT_EVIDLIST_TO_NAME" (P1, P2, P3, P4, P5)
```

Where:

Table 50. Function Parameters

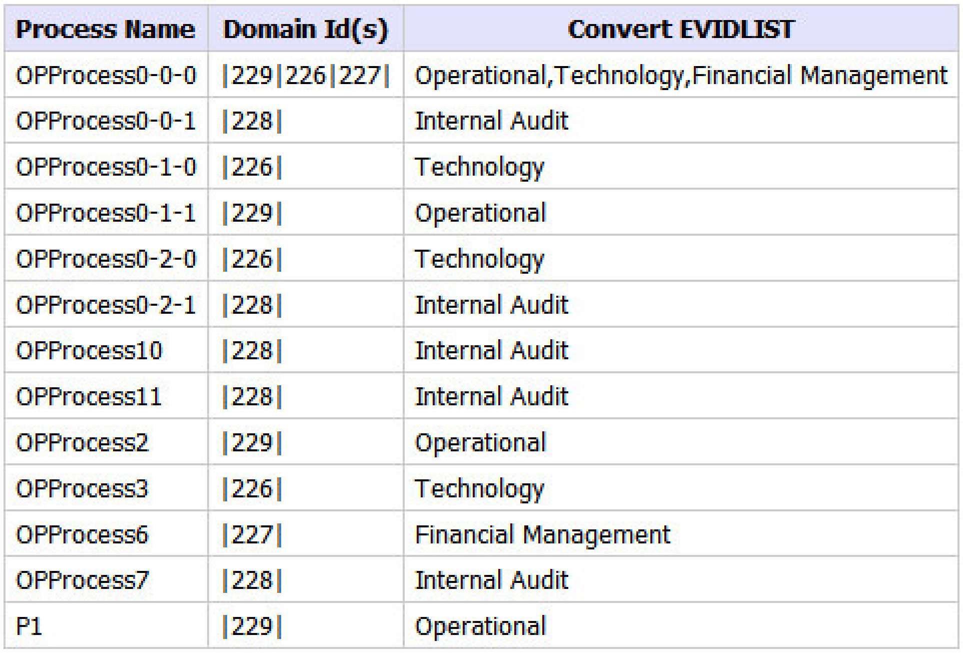
Parameter	Description
P1	Required. The variable name of the multivalue enumerated variable to be converted to a string.
P2	Required. The locale ID (a numeric identifier). Used to automatically translate the text of enumerated strings to the locale of the user running the report. **Note:** A value of `NULL` (without quotes) can be used; however, providing the locale session parameter or locale ID provides the best processing performance. Using `NULL` will force the function to use the system-wide default locale ID. To return the session parameter that gives the value of the locale of the user running the report, use the following expression: `#$account.parameters.openPagesLocaleId#` Default: `NULL` (without quotes and not case sensitive)
P3	Optional. Specifies the delimiter character to be used as a separator for items in selection lists with multiple values (multi-valued enumerations). Typically this value may be a comma (,), semi-colon (;), or pipe (|) character. Default: `','`
P4	Optional. Specifies the column on which the selection list values (multi-valued enumerations) will be sorted. The possible values are: • `'EV_DISPLAY_ORDER'` - use to sort the list values by the order of their enumerated value display as they appear in the application user interface. • `'EV_NAME'` - use to sort the list values by the localized enumerated value name. Default: `'EV_DISPLAY_ORDER'`

Table 50. Function Parameters (continued)

Parameter	Description
P5	Optional. Specifies the sort direction for items in selection lists with multiple values (multi-valued enumerations). The possible values are: • 'ASC' - this sorts the list values in ascending order. • 'DESC' - this sorts the list values in descending order. Default: 'ASC'

Usage Example

The following example shows how to take the Domain field from the RT_PROCESS table and convert it to a text string list. Figure 65 shows sample report output.



Process Name	Domain Id(s)	Convert EVIDLIST
OPProcess0-0-0	|229|226|227|	Operational,Technology,Financial Management
OPProcess0-0-1	|228|	Internal Audit
OPProcess0-1-0	|226|	Technology
OPProcess0-1-1	|229|	Operational
OPProcess0-2-0	|226|	Technology
OPProcess0-2-1	|228|	Internal Audit
OPProcess10	|228|	Internal Audit
OPProcess11	|228|	Internal Audit
OPProcess2	|229|	Operational
OPProcess3	|226|	Technology
OPProcess6	|227|	Financial Management
OPProcess7	|228|	Internal Audit
P1	|229|	Operational

Figure 65. Convert EVIDLIST to Name Output

Converting Single Value Enumerations to Names

This function converts a single enumerated value ID into a localized name.

Report authors can use this function to turn an enumeration ID into a name. This function eliminates the need to use custom SQL statements or additional query subjects.

Syntax

This function has the following syntax:

```
"OP_RPS_AUX.CONVERT_EVID_TO_NAME" (P1, P2)
```

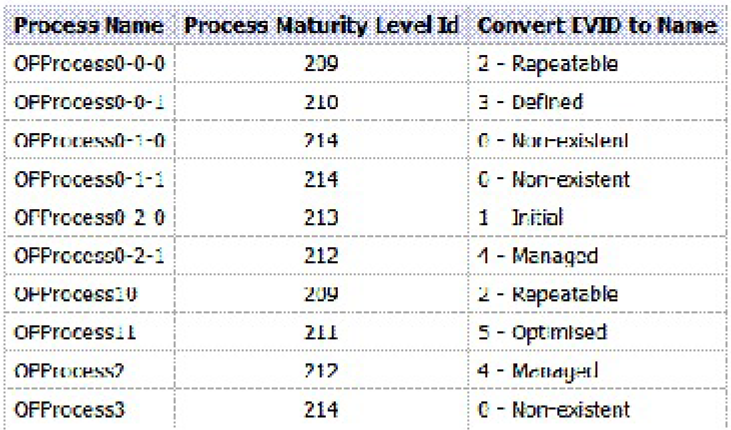
Where:

Table 51. Function Parameters

Parameter	Description
P1	Required. The numeric ID of the single value enumerated variable to be converted to a string.
P2	Required. The locale ID (a numeric identifier). Used to automatically translate the text of enumerated strings to the locale of the user running the report. **Note:** A value of `NULL` (without quotes) can be used; however, providing the locale session parameter or locale ID provides the best processing performance. Using `NULL` will force the function to use the system-wide default locale ID. To return the session parameter that gives the value of the locale of the user running the report, use the following expression: `#$account.parameters.openPagesLocaleId#` Default: `NULL` (without quotes and not case sensitive)

Usage Example

The following example shows how to take the Process Maturity Level ID from the SOXProcess object and convert it to its text string. Figure 66 shows sample report output.



Process Name	Process Maturity Level Id	Convert EVID to Name
OPProcess0-0-0	209	2 - Repeatable
OPProcess0-0-1	210	3 - Defined
OPProcess0-1-0	214	0 - Non-existent
OPProcess0-1-1	214	0 - Non-existent
OPProcess0 2 0	213	1 Initial
OPProcess0-2-1	212	4 - Managed
OPProcess10	209	2 - Repeatable
OPProcess11	211	5 - Optimised
OPProcess2	212	4 - Managed
OPProcess3	214	0 - Non-existent

Figure 66. Convert EVID to Name Output

Counting the Number of Text Occurrences

This function returns the count of a specified text in a given string.

Syntax

This function has the following syntax:

```
"OP_UTILITIES.COUNT_TEXT_OCCURENCE" (P1, P2)
```

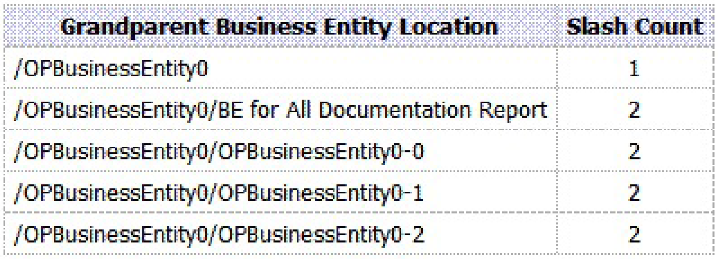
Where:

Table 52. Function Parameters

Parameter	Description
P1	Required. The text string to be evaluated.
P2	Required. The string to be counted.

Usage Example

The following example counts the number of forward slashes (/) in the full path of a grandparent business entity. Figure 67 shows sample report output.



Grandparent Business Entity Location	Slash Count
/OPBusinessEntity0	1
/OPBusinessEntity0/BE for All Documentation Report	2
/OPBusinessEntity0/OPBusinessEntity0-0	2
/OPBusinessEntity0/OPBusinessEntity0-1	2
/OPBusinessEntity0/OPBusinessEntity0-2	2

Figure 67. Count Text Occurrence Output

Determining a Framework Relationship Between Objects

This function identifies whether or not a parent-child relationship exists (either direct or indirect, primary or non-primary) between any two objects in a dimensional framework. This function is specifically used in the dimensional model in the IBM OpenPages framework to join standalone objects when required by the user. It eliminates duplicate counting of rolled up objects and properly handles data that is pushed down from a higher level in the hierarchy.

Syntax

This function has the following syntax:

```
"OP_RPS_AUX.IS_FM_REL_PARENT_CHILD" (P1, P2, P3, P4, P5, P6, P7, P8, P9,
P10)
```

Where:

Table 53. Function Parameters

Parameter	Description
P1	Required. The object ID (a numeric value) of the parent object type that is higher in the object hierarchy than the object provided in the *P2* parameter.
P2	Required. The object ID (a numeric value) of the child object type that is lower in the object hierarchy than the object provided in the *P1* parameter.
P3	Required. The reporting period ID (a numeric value). Default: -1 (the numeric identifier for the current reporting period) **Note:** The value of the reporting period ID can also be passed as a prompt parameter variable (for example, `?rpid?` where `rpid` represents a variable name). Providing a value will optimize report performance.

Table 53. Function Parameters (continued)

Parameter	Description
P4	Required. The hierarchy depth based on the Parent Resource Object Type (i.e. number of levels in the hierarchy). This number would be 1 for non-recursive Object Type hierarchies, and 1 or greater for recursive ones. Default: 1
P5	Required. The "OP Native Level" of the Parent Hierarchy Root Resource. This number would be 1 for all non-Entity Object Type Hierarchies, and 1 or greater for Entity-based ones. Default: 1
P6	Required. The "OP Native Level" of the Parent Resource. Default: 1
P7	Required. Indicator for the type of parent-child relationship to look for between the objects designated in the *P1* and *P2* parameters. The possible values are: • `'N'` - checks for any relationship (primary or non-primary) between objects. • `'Y'` - checks for a primary-only relationship between objects. Default: 'N'
P8	Optional. Indicator that directs how the SQL engine should execute its ConnectBy Statement. The possible values are: • `'Y'` - Executes the ConnectBy statement from bottom to top (bottom up). • `'N'` - Executes the ConnectBy statement from top to bottom (top down). **Note:** This parameter is ignored if the *P4* parameter is set to primary-only relationship. Default: 'Y'
P9	Optional. Defines the result if any of the resources (parent or child) are null. The possible values are: • `'Y'` - returns a 'Y' when either parent or child objects are null. • `'N'` - returns a 'N' when either parent or child objects are null. Default: 'N'
P10	Optional. The numeric set identifier. Default: 1 (always use the default value)

Usage Example

The following sample function is used in a join between an entity query subject and a standalone issue query subject to obtain all issues related directly or indirectly to an entity. This function was used because the sub-report that contained this join was called from a dimensional report and the quantities in the two reports needed to match.

"OP_RPS_AUX.IS_FM_REL_PARENT_CHILD"([Entity].[LOB FUNCTION_ENTITY_ID], [Issue].[IS_ISSUE_ID], [Entity].[LOB FUNCTION_REPORTING_PERIOD_ID], 3, 1, [Entity].[Native Level]) = 'Y'

The value [Entity].[Native Level] was passed from the calling report as a parameter. Since this involved a fixed entity hierarchy the hierarchy depth and native level were hard coded.

Determining a Primary Parent Relationship Between Objects

This function identifies whether a relationship (either direct or indirect) between any two objects is primary or not. Report authors can use this function, for example, to display in a report which object is the primary parent for child objects with multiple parents, or as a filter for returning details on only those objects with a primary relationship.

Syntax

This function has the following syntax:

```
"OP_RPS_AUX.IS_REL_PARENT_CHILD_PRIM" (P1, P2, P3, P4, P5, P6, P7, P8, P9,
P10)
```

Where:

Table 54. Function Parameters

Parameter	Description
P1	Required. The object ID (a numeric value) of the parent object. This object type must be higher in the object hierarchy than the child object provided in the *P2* parameter.
P2	Required. The object ID (a numeric value) of the child object. This object type must be lower in the object hierarchy than the parent object provided in the *P1* parameter.
P3	Required. The reporting period ID (a numeric value) for which the primary parent attribute will be returned. **Note:** The value of the reporting period ID can also be passed as a prompt parameter variable (for example, `?rpid?` where `rpid` represents a variable name). Providing a value will optimize report performance. Default: -1 (the numeric identifier for the current reporting period)
P4	Required. The object type ID (a numeric value) of the parent object. This value will be looked up dynamically for direct relationships. Default: `NULL` (without quotes and not case sensitive; always use the default value)
P5	Required. The object type ID (a numeric value) of the child object. This value will be looked up dynamically for direct relationships. Default: `NULL` (without quotes and not case sensitive; always use the default value)
P6	Required. indicator to always use Connect By SQL. Default: ‘N’ (always use the default value)

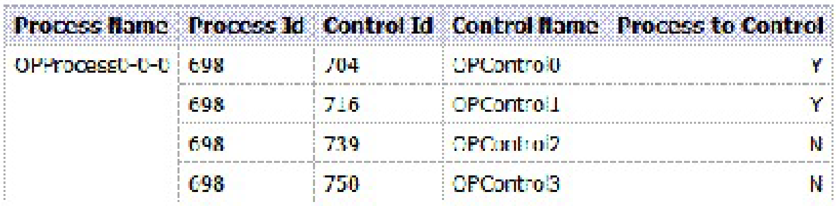
Table 54. Function Parameters (continued)

Parameter	Description
P7	Required. indicator that identifies whether a relationship (either direct or indirect) between any two resources is primary or not. The possible values are: • 'Y' - returns a 'Y' when two objects have a direct and primary relationship (for example, between Test and Test Result objects). • 'N' - returns a 'Y' if the child object can trace a primary relationship through the object hierarchy to the parent object. Use this setting when the specified objects have an indirect parent-child type relationship within the object hierarchy (for example, Process and Control objects have an indirect relationship where Control Objective and Risk objects set between Process and Control in the hierarchy). Default: 'Y'
P8	Required. Reserved function. Default: NULL (without quotes and not case sensitive; always use the default value)
P9	Required. Request the result if any of the resources (parent or child) are null. The possible values are: • 'Y' - returns a 'Y' when the relationship between two objects is primary. • 'N' - returns a 'N' when the relationship between two objects is not primary. Default: 'Y'
P10	Optional. The numeric set identifier. Default: 1 (always use the default value)

Usage Example

The following sample function returns a 'Y' or 'N' in a row for a particular Control if that Control can trace a primary relationship up the hierarchy through Risk, through Control Objective, and finally up to Process within the current Reporting Period.

The "process to control" calculated column in Figure 68 returns the information indicating whether or not there is a primary relationship between a given Process and Control object.



Process Name	Process Id	Control Id	Control Name	Process to Control
OPProcess0-0-0	698	704	OPControl0	Y
	698	716	OPControl1	Y
	698	739	OPControl2	N
	698	750	OPControl3	N

Figure 68. Is Relative Parent Child Primary Output

Adding a filter for the *P7* parameter, such as changing 'N' to 'Y' for process to control, will remove all row details where there is not a primary relationship (path) between a given Process and Control object.

Determining a Relationship Between Objects

This function identifies whether or not a parent-child relationship exists (either direct or indirect, primary or non-primary) between any two objects.

Syntax

This function has the following syntax:

```
"OP_RPS_AUX.IS_REL_PARENT_CHILD" (P1, P2, P3, P4, P5, P6, P7)
```

Where:

Table 55. Function Parameters

Parameter	Description
P1	Required. The object ID (a numeric value) of the parent object type that is higher in the object hierarchy than the object provided in the *P2* parameter.
P2	Required. The object ID (a numeric value) of the child object type that is lower in the object hierarchy than the object provided in the *P1* parameter.
P3	Required. The reporting period ID (a numeric value). Default: -1 (the numeric identifier for the current reporting period) **Note:** The value of the reporting period ID can also be passed as a prompt parameter variable (for example, `?rpid?` where `rpid` represents a variable name). Providing a value will optimize report performance.
P4	Optional. Indicator for the type of parent-child relationship to look for between the objects designated in the *P1* and *P2* parameters. The possible values are: • `'N'` - checks for any relationship (primary or non-primary) between objects. • `'Y'` - checks for a primary-only relationship between objects. Default: 'N'
P5	Optional. Indicator that directs how the SQL engine should execute its ConnectBy Statement. The possible values are: • `'Y'` - Executes the ConnectBy statement from bottom to top (bottom up). • `'N'` - Executes the ConnectBy statement from top to bottom (top down). **Note:** This parameter is ignored if the *P4* parameter is set to primary-only relationship. Default: 'Y'
P6	Optional. Defines the result if any of the resources (parent or child) are null. The possible values are: • `'Y'` - returns a 'Y' when either parent or child objects are null. • `'N'` - returns a 'N' when either parent or child objects are null. Default: 'N'

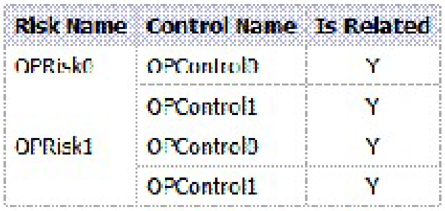
Table 55. Function Parameters (continued)

Parameter	Description
P7	Optional. The numeric set identifier. Default: 1 (always use the default value)

Usage Example

The following sample function indicates whether or not any kind of relationship exists between the Risk record and the Control record (primary or non-primary, direct or indirect).

The "Is Related" column in Figure 69 returns the information indicating whether or not there is a parent-child relationship between a given Risk and Control object.



Risk Name	Control Name	Is Related
OPRisk0	OPControl0	Y
	OPControl1	Y
OPRisk1	OPControl0	Y
	OPControl1	Y

Figure 69. Is Relative Parent Child Output

Formatting the Currency Value

This function formats a currency value with the proper number of decimal places and separators for the specified locale.

This function can be useful for computed fields.

Note: The currency symbol is NOT displayed in the output.

Syntax

This function has the following syntax:

```
"OP_CURRENCY_MGR.FORMAT_CURRENCY_VALUE" (P1, P2, P3)
```

Where:

Table 56. Function Parameters

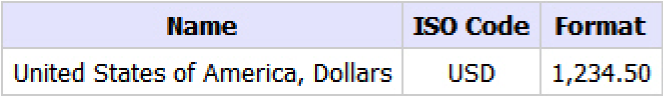
Parameter	Description
P1	Required. The number (such as the currency amount) you want formatted.
P2	Required. The 3-character ISO currency code that determines the number of decimal places to be displayed.

Table 56. Function Parameters (continued)

Parameter	Description
P3	Required. The numeric identifier of the OpenPages GRC on Cloud user locale that determines the type of group separators. **Note:** A value of `NULL` (without quotes) can be used; however, providing the locale session parameter or locale ID provides the best processing performance. Using `NULL` will force the function to use the system-wide default locale ID. To return the session parameter that gives the value of the locale of the user running the report, use the following expression: `#$account.parameters.openPagesLocaleId#` Default: `NULL` (without quotes and not case sensitive)

Usage Example 1

The following example translates the value for 1234.5 into a format for the US locale.

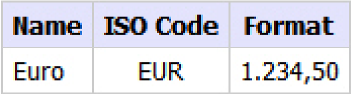


Name	ISO Code	Format
United States of America, Dollars	USD	1,234.50

Figure 70. Format Currency for USD

Usage Example 2

The following example translates the value for 1234.5 into a format for the German locale. Figure 71 shows sample report output.

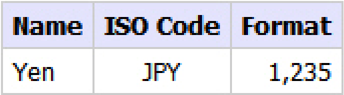


Name	ISO Code	Format
Euro	EUR	1.234,50

Figure 71. Format Currency for EUR

Usage Example 3

The following example translates the value for 1234.5 into a format for the Japanese locale. Figure 72 shows sample report output



Name	ISO Code	Format
Yen	JPY	1,235

Figure 72. Format Currency for JPY

Getting an Actor’s Display Name by ID

This function applies only to Oracle environments, and returns an actor's (user or group) formatted display name using the internal Actor ID number.

If the actor is a group, only the group’s account name is returned, and the supplied format string is ignored. If an invalid actor name is supplied, the supplied value is returned.

Syntax

This function has the following syntax:

```
"OP_ACTOR_MGR.GET_DISPLAY_NAME" (P1, P2, P3)
```

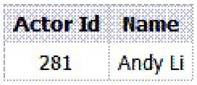
Where:

Table 57. Function Parameters

Parameter	Description
P1	Required. A numeric ID for the actor (user or group) to be displayed.
P2	Required. The locale ID (a numeric identifier). Used to automatically translate the text of enumerated strings to the locale of the user running the report. **Note:** A value of `NULL` (without quotes) can be used; however, providing the locale session parameter or locale ID provides the best processing performance. Using `NULL` will force the function to use the system-wide default locale ID. To return the session parameter that gives the value of the locale of the user running the report, use the following expression: `#$account.parameters.openPagesLocaleId#` Default: `NULL` (without quotes and not case sensitive)
P3	Optional. The format display options are: • `%FN;` to display the user's first name • `%LN;` to display the user's last name • `%EM;` to display the user's email address Default: The logon account name of the actor is displayed if no format is specified.

Usage Example 1

The following example displays the first and last name for actor ID 281. Figure 73 shows sample report output.

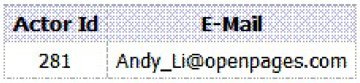


Actor Id	Name
281	Andy Li

Figure 73. Display Name by ID First and Last Name

Usage Example 2

The following example displays the email for actor ID 281. Figure 74 shows sample report output.

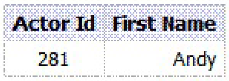


Actor Id	E-Mail
281	Andy_Li@openpages.com

Figure 74. Display Name by ID Email

Usage Example 3

The following example displays the first name for actor ID 281. Figure 75 shows sample report output.



Actor Id	First Name
281	Andy

Figure 75. Display Name by ID First Name

Getting an Actor’s Display Name by Actor ID

This function applies only to IBM DB2 environments, and returns an actor's (user or group) formatted display name by using the internal Actor ID number.

If the actor is a group, only the group’s account name is returned, and the supplied format string is ignored. If an invalid actor name is supplied, the supplied value is returned.

Syntax

This function has the following syntax:

```
"OP_ACTOR_MGR.GET_DISPLAY_NAME_BY_ACTOR_ID" (P1, P2, P3)
```

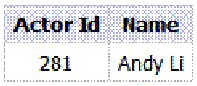
Where:

Table 58. Function Parameters

Parameter	Description
P1	Required. A numeric ID for the actor (user or group) to be displayed.
P2	Required. The locale ID (a numeric identifier). Used to automatically translate the text of enumerated strings to the locale of the user running the report. **Note:** A value of `NULL` (without quotes) can be used; however, providing the locale session parameter or locale ID provides the best processing performance. Using `NULL` will force the function to use the system-wide default locale ID. To return the session parameter that gives the value of the locale of the user running the report, use the following expression: `#$account.parameters.openPagesLocaleId#` Default: `NULL` (without quotes and not case sensitive)
P3	Optional. The format display options are: • `%FN`; to display the user’s first name • `%LN`; to display the user’s last name • `%EM`; to display the user’s email address Default: The logon account name of the actor is displayed if no format is specified.

Usage Example 1

The following example displays the first and last name for actor ID 281. Figure 76 shows sample report output.

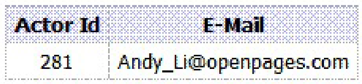


Actor Id	Name
281	Andy Li

Figure 76. Display Name by ID First and Last Name

Usage Example 2

The following example displays the email for actor ID 281. Figure 77 shows sample report output.

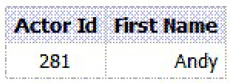


Actor Id	E-Mail
281	Andy_Li@openpages.com

Figure 77. Display Name by ID Email

Usage Example 3

The following example displays the first name for actor ID 281. Figure 78 shows sample report output.



Actor Id	First Name
281	Andy

Figure 78. Display Name by ID First Name

Getting an Actor’s Display Name by Logon ID

This function returns actor's (users or groups) formatted display names.

If the actors are a group, only the group account names are returned, and the supplied format string is ignored. If an invalid actor Login ID is supplied, the supplied value is returned.

Syntax

This function has the following syntax:

```
"OP_ACTOR_MGR.GET_DISPLAY_NAME" (P1, P2, P3, P4)
```

Where:

Table 59. Function Parameters

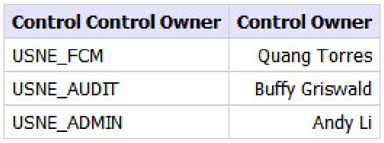
Parameter	Description
P1	Required. The string value logon user names for the actors (users or groups) to be displayed.

Table 59. Function Parameters (continued)

Parameter	Description
P2	Required. The locale ID (a numeric identifier). Used to automatically translate the text of enumerated strings to the locale of the user running the report. **Note:** A value of NULL (without quotes) can be used; however, providing the locale session parameter or locale ID provides the best processing performance. Using NULL will force the function to use the system-wide default locale ID. To return the session parameter that gives the value of the locale of the user running the report, use the following expression: #$account.parameters.openPagesLocaleId# Default: NULL (without quotes and not case sensitive)
P3	Optional. The format display options are: • %FN; to display the user's first name • %LN; to display the user's last name • %EM; to display the user's email address Default: The logon ID of the actor is displayed if no format is specified.
P4	Optional. Separator for returned display names. Default: A comma followed by a space.

Usage Example 1

The following example displays the first and last name of users who are designated as a 'Control Owner' in the application. Figure 79 shows sample report output.

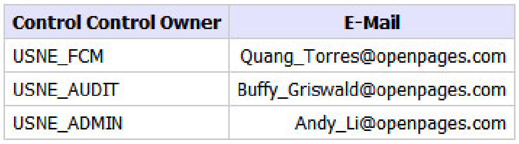


Control Control Owner	Control Owner
USNE_FCM	Quang Torres
USNE_AUDIT	Buffy Griswald
USNE_ADMIN	Andy Li

Figure 79. Control Owner First and Last Name

Usage Example 2

The following example displays the email of users who are designated as a 'Control Owner' in the application. Figure 80 shows sample report output.

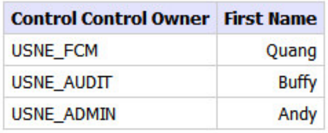


Control Control Owner	E-Mail
USNE_FCM	Quang_Torres@openpages.com
USNE_AUDIT	Buffy_Griswald@openpages.com
USNE_ADMIN	Andy_Li@openpages.com

Figure 80. Control Owner Email

Usage Example 3

The following example displays the first name of users who are designated as a 'Control Owner' in the application. Figure 81 on page 175 shows sample report output.



Control Control Owner	First Name
USNE_FCM	Quang
USNE_AUDIT	Buffy
USNE_ADMIN	Andy

Figure 81. Control Owner First Name

Getting External Key Text

This function returns a localized external key string.

This function can be useful for localizing computed fields with a data type of 'Simple String' as these type of fields are not localized by the application. When localized strings are needed for a computed field (excluding field label names, which are already localized) a computed field author can add a new application text key through the IBM OpenPages GRC on Cloud user interface and then reference it with this function. In this way, you can manage all the localized text in a single place and avoid coding locale specific logic into equations.

Syntax

This function has the following syntax:

```
"OP_I18N_MGR.GET_EXTERNAL_KEY_TEXT" (P1, P2, P3)
```

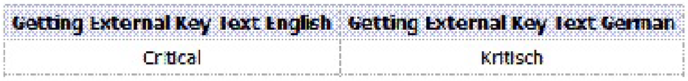
Where:

Table 60. Function Parameters

Parameter	Description
P1	Required. The name of the application text key.
P2	Required. The application text category.
P3	Required. The numeric identifier of the OpenPages GRC on Cloud user locale. **Note:** A value of `NULL` (without quotes) can be used; however, providing the locale session parameter or locale ID provides the best processing performance. Using `NULL` will force the function to use the system-wide default locale ID. To return the session parameter that gives the value of the locale of the user running the report, use the following expression: `#$account.parameters.openPagesLocaleId#` Default: `NULL` (without quotes and not case sensitive)

Usage Example

The following example shows how to take the risk rating value of ‘critical’ and substitute a localized string value for it. Figure 82 on page 176 shows sample report output.



Getting External Key Text English	Getting External Key Text German
Critical	Kritisch

Figure 82. Getting External Key Text Output

Getting the Attributes of a Primary Parent Object

This function returns the value for one or more attributes of a primary parent object.

Report authors can use this function, for example, to directly look up any ("regular" or enumerated single-valued or multivalued) property field value for a given object without having to query through the entire object hierarchy. You must be very careful in using this function in situations where an object can have multiple parents.

Note: Please also review the Getting the Direct Attributes of a Primary Parent Object before using this function because the latter function is easier to use.

Syntax

This function has the following syntax:

```
"OP_RPS_AUX.GET_PRIM_PARENT_ATTRIBUTE" (P1, P2, P3, P4, P5, P6, P7, P8, P9,
P10, P11, P12, P13, P14)
```

Where:

Table 61. Function Parameters

Parameter	Description
P1	Required. The object ID (a numeric value) of the child object for which the primary parent attributes will be returned.
P2	Required. The reporting period ID (a numeric value) of the child object for which the primary parent attributes will be returned. **Note:** The value of the reporting period ID can also be passed as a prompt parameter variable (for example, `?rpid?` where `rpid` represents a variable name). Providing a value will optimize report performance. Default: -1 (the numeric identifier for the current reporting period)
P3	Required. The object type ID of the parent object (this numeric value is different from the object ID value in the *P1* parameter). For example, if the parent object is a Business Entity, it is the object type ID of the Business Entity (`SOXBusEntity`) object type. **Note:** Reports may not be portable between installations (as same object types may have different IDs)

Table 61. Function Parameters (continued)

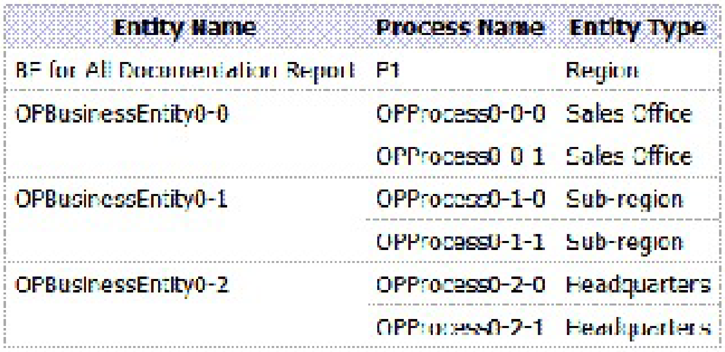
Parameter	Description
P4	Required. The level of the primary parent object in the hierarchy. Used only for recursive objects such as: Business Entity, Sub-Process, Sub-Mandate, Sub-Account. The possible values are: • 1 (the highest level parent) • 0 (lowest level parent closest to the object) • 4 (fourth level parent from the top. It does not exist, use level 0) • -1 (second lowest level parent from the bottom) • -4 (fourth level parent from the bottom. It does not exist, use level 1) You can specify any level as required. If a specified value does not exist in the hierarchy, the highest level parent object will be used. **Note:** If the parent object type is non-recursive use 1. Default: 1 (highest level parent)
P5	Required. The RT table name of the parent object type in the Reporting Schema. **Note:** This value is NOT the query subject name that a report author sees in Report Studio. A report author should be able to determine the names of the tables by connecting to the database. Default: `'RT_ENTITY'`
P6	Required. The object ID column name in the RT table of the parent object type in the Reporting Schema. **Note:** This is NOT the query item name that the report author sees in Report Studio. A report author should be able to determine the names of the columns by connecting to the database. Default: `'ENTITY_ID'`
P7	Required. The reporting period ID column name in the RT table of the parent object type in the Reporting Schema. **Note:** This is NOT the query item name that the report author sees in Report Studio. A report author should be able to determine the names of the columns by connecting to the database. Default: `'REPORTING_PERIOD_ID'`
P8	Required. The column name of the attribute in the RT table of the parent object type in the Reporting Schema. This represents the data item that will be output back to the report or filter. **Note:** This is NOT the query item name that the report author sees in Report Studio. A report author should be able to determine the names of the columns by connecting to the database. Default: `'FULL_PATH'`
P9	Required. Specifies the attribute type for values returned in the *P8* column parameter. **Note:** The values are not case sensitive. The possible values are: • `'NONE'` (use for ordinary (non-enumerated) fields) • `'SINGLE'` (use for single-valued enumerations) • `'MULTI'` (use for multivalued enumerations) Default: `'NONE'`

Table 61. Function Parameters (continued)

Parameter	Description
P10	Required. The locale ID (a numeric identifier). Used to automatically translate the text of enumerated strings to the locale of the user running the report. **Note:** A value of `NULL` (without quotes) can be used; however, providing the locale session parameter or locale ID provides the best processing performance. Using `NULL` will force the function to use the system-wide default locale ID. To return the session parameter that gives the value of the locale of the user running the report, use the following expression: `#$account.parameters.openPagesLocaleId#` Default: `NULL` (without quotes and not case sensitive)
P11	Optional. Specifies the delimiter character to be used as a separator for items in selection lists with multiple values (multi-valued enumerations). Typically this value may be a comma (,), semi-colon (;), or pipe (|) character. Default: `','`
P12	Optional. Specifies the column on which the selection list values (multi-valued enumerations) will be sorted. The possible values are: • `'EV_DISPLAY_ORDER'` - use to sort the list values by the order of their enumerated value display as they appear in the application user interface. • `'EV_NAME'` - use to sort the list values by the localized enumerated value name. Default: `'EV_DISPLAY_ORDER'`
P13	Optional. Specifies the sort direction for items in selection lists with multiple values (multi-valued enumerations). The possible values are: • `'ASC'` - this sorts the list values in ascending order. • `'DESC'` - this sorts the list values in descending order. Default: `'ASC'`
P14	Optional. The numeric set identifier. Default: 1 (always use the default value)

Usage Example 1

The following example shows the values of the attribute 'ENTITY_TYPE' for the parent entity of a process. Figure 83 on page 179 shows sample report output.



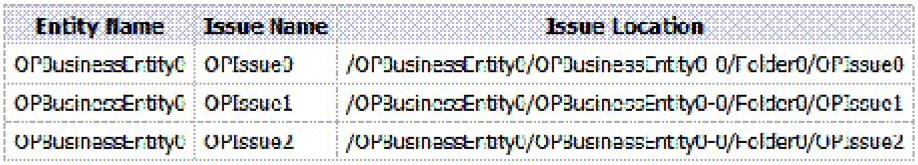
Entity Name	Process Name	Entity Type
BF for All Documentation Report	P1	Region
OPBusinessEntity0-0	OPProcess0-0-0	Sales Office
	OPProcess0 0 1	Sales Office
OPBusinessEntity0-1	OPProcess0-1-0	Sub-region
	OPProcess0-1-1	Sub-region
OPBusinessEntity0-2	OPProcess0-2-0	Headquarters
	OPProcess0-2-1	Headquarters

Figure 83. Get Primary Parent Attribute Output

Usage Example 2

By using the OP_RPS_AUX.GET_PRIM_PARENT_ATTRIBUTE function, a report could be authored solely from the Issue query subject without having to add in additional objects. The query generated by the report would not, in this example, need to traverse the full hierarchy, and would still return all required data items to the report output or filter. The result of this usage is that it may be possible to dramatically improve report runtime.

The following example returns the top-level 'Entity Name' for a given Issue ID. Figure 84 shows sample report output.



Entity Name	Issue Name	Issue Location
OPBusinessEntity0	OPIssue0	/OPBusinessEntity0/OPBusinessEntity0 0/Folder0/OPIssue0
OPBusinessEntity0	OPIssue1	/OPBusinessEntity0/OPBusinessEntity0-0/Folder0/OPIssue1
OPBusinessEntity0	OPIssue2	/OPBusinessEntity0/OPBusinessEntity0-0/Folder0/OPIssue2

Figure 84. Get Primary Parent Attribute Output

Getting the Direct Attributes of a Primary Parent Object

This function returns the value for an attribute of a primary parent object.

Report authors can use this function to directly look up certain property values for a given object without having to query through the entire object hierarchy. This function does not require the report author to know what type of primary parent object they are querying.

Syntax

This function has the following syntax:

```
"OP_RPS_AUX.GET_DIRECT_PRIM_PRNT_VALUE" (P1, P2, P3, P4, P5)
```

Where:

Table 62. Function Parameters

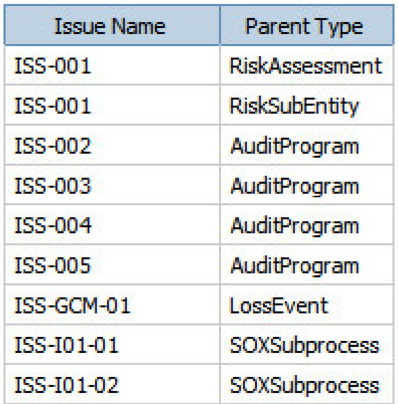
Parameter	Description
P1	Required. The object ID (a numeric value) of the child object for which the primary parent attributes will be returned.

Table 62. Function Parameters (continued)

Parameter	Description
P2	Required. The reporting period ID (a numeric value) for which the value will be returned. Default: `NULL` (without quotes and not case sensitive) returns the current reporting period ID, which is -1. Providing a value will optimize report performance.
P3	Required. The actor ID of the user that is logged in. This is used to determine what data access the user possesses.
P4	Required. The property name of the attribute of the parent object type in the Reporting Schema. This represents the data item that will be output back to the report or filter. The following is the list of valid property names: • RESOURCE_ID - The unique system ID of the resource • RESOURCE_NAME - The parent name • RESOURCE_NAME_RAW - The entire parent name with internal identifiers • RESOURCE_LOCATION - The parent full path • RESOURCE_LOCATION_RAW - The entire parent full path without the internal identifiers stripped off • RESOURCE_URL - The parent detail page URL • OBJECT_TYPE_ID - The resource ID of the parent object • OBJECT_TYPE_NAME_SYS - The parent system object name • OBJECT_TYPE_NAME_I18N - The parent localized object name
P5	Required. The locale ID (a numeric identifier). Used to automatically translate the text of enumerated strings to the locale of the user running the report. **Note:** A value of `NULL` (without quotes) can be used; however, providing the locale session parameter or locale ID provides the best processing performance. Using `NULL` will force the function to use the system-wide default locale ID. To return the session parameter that gives the value of the locale of the user running the report, use the following expression: `#$account.parameters.openPagesLocaleId#` Default: `NULL` (without quotes and not case sensitive)

Usage Example 1

The following example returns the parent type for a given issue ID. Figure 85 on page 181 shows sample report output.

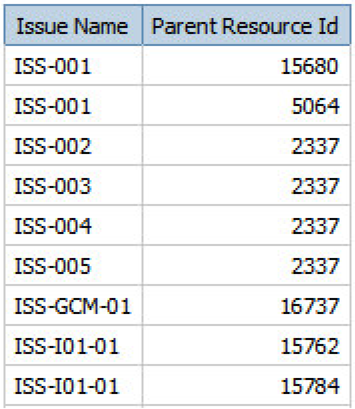


Issue Name	Parent Type
ISS-001	RiskAssessment
ISS-001	RiskSubEntity
ISS-002	AuditProgram
ISS-003	AuditProgram
ISS-004	AuditProgram
ISS-005	AuditProgram
ISS-GCM-01	LossEvent
ISS-I01-01	SOXSubprocess
ISS-I01-02	SOXSubprocess

Figure 85. Primary Parent Type

Usage Example 2

The following example returns the parent resource ID for a given issue ID. Figure 86 shows sample report output.



Issue Name	Parent Resource Id
ISS-001	15680
ISS-001	5064
ISS-002	2337
ISS-003	2337
ISS-004	2337
ISS-005	2337
ISS-GCM-01	16737
ISS-I01-01	15762
ISS-I01-01	15784

Figure 86. Primary Parent ID

Getting the Exchange Rate

This function returns the exchange rate for the specified ISO currency code and date.

If no date is specified, the current system date is used. This function can be useful for computed fields.

Syntax

This function has the following syntax:

```
"OP_CURRENCY_MGR.GET_EX_RATE_ISO" (P1, P2)
```

Where:

Table 63. Function Parameters

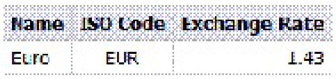
Parameter	Description
P1	Required. The 3-character ISO currency code.

Table 63. Function Parameters (continued)

Parameter	Description
P2	Optional. The date the exchange rate became effective in the system. Default: The current system date is used if no date is specified.

Usage Example 1

The following example returns the current exchange rate for Euros. Figure 87 shows sample report output.

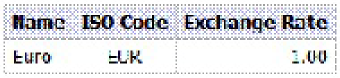


Name	ISO Code	Exchange Rate
Euro	EUR	1.43

Figure 87. Currency Exchange Rate

Usage Example 2

The following example returns the exchange rate for Euros for the date it was last changed, January 1, 1970. Figure 86 on page 181 shows sample report output.



Name	ISO Code	Exchange Rate
Euro	EUR	1.00

Figure 88. Currency Exchange Rate by Date

Getting Tree Path Resource IDs

This function returns the IDs of the objects that are in the path of two objects that are provided as inputs to the function.

Syntax

This function has the following syntax:

```
"OP_RESOURCES_MGR.GET_TREE_PATH_RESOURCE_IDS" (P1, P2, P3, P4, P5)
```

Where:

Table 64. Function Parameters

Parameter	Description
P1	Required. The resource identifier for the top of your tree segment.
P2	Required. The resource identifier for the bottom of your tree segment.
P3	Required. A comma separated, parenthesis enclosed list of object types to be visited.
P4	Optional. The reporting period ID. Default: The current reporting is used if no reporting period is specified. **Note:** The value of the reporting period ID can also be passed as a prompt parameter variable (for example, `?rpid?` where `rpid` represents a variable name). Providing a value will optimize report performance.

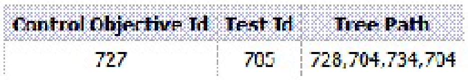
Table 64. Function Parameters (continued)

Parameter	Description
P5	Optional. The hierarchy ID. Defaults to the currently supported hierarchy. Default: 1 (always use the default value)

Usage Example

The following example shows that ?ControlObjectiveID? (id_start) and ?TestID? (id_end) are parameters to the report for the listed objects (SOXControlObjective, SOXRisk, SOXControl, and SOXTest). Figure 89 shows sample report output.

```
"OP_RESOURCES_MGR.GET_TREE_PATH_RESOURCE_IDS"(?ControlObjectiveID?,?TestID?,
'(' || '''SOXControlObjective''' || ',' || '''SOXRisk''' || ',' ||
 '''SOXControl''' || ',' || '''SOXTest''' || ')' )
```



Control Objective Id	Test Id	Tree Path
727	705	728,704,734,704

Figure 89. Getting Tree Path Output

Other results might be:

- `-1` if only a direct path between resources exists
- `-2` if no path exists between resources

Rolling up Multivalued Enumerations Into a List

This function returns a delimited list of localized multivalued enumeration names for a specified multivalued enumeration resource property field.

Report authors can use this function to roll up multivalue enumerations into a comma (or other desired delimiter) separated list. This function eliminates the need to use custom SQL statements or a Master Detail approach.

Syntax

This function has the following syntax:

```
"OP_RPS_AUX.GET_MVAL_STRAGG_GENERIC" (P1, P2, P3, P4, P5, P6, P7, P8, P9,
P10)
```

Where:

Table 65. Function Parameters

Parameter	Description
P1	Required. The RT table name that stores the multivalued enumerated values for a given property of a given object type
P2	Required. The column name of the RT table specified in P1 that stores the resource ID.
P3	Required. The column name of the RT table specified in P1 that stores the enumerated value IDs.

Table 65. Function Parameters (continued)

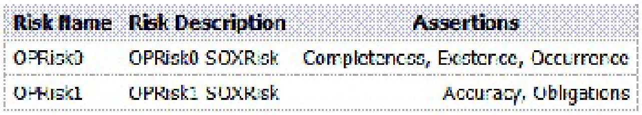
Parameter	Description
P4	Required. The column name of the RT table specified in P1 that stores the reporting period ID.
P5	Required. The resource ID (a numeric value) for which the list will be returned. It is the Query Item from the Query Subject that contains the main resource ID by which the enumerated values will be aggregated.
P6	Required. The reporting period ID (a numeric value) for which the list will be returned. Default: `NULL` (without quotes and not case sensitive) returns the current reporting period ID, which is -1. Providing a value will optimize report performance.
P7	Required. The locale ID (a numeric identifier). Used to automatically translate the text of enumerated strings to the locale of the user running the report. **Note:** A value of `NULL` (without quotes) can be used; however, providing the locale session parameter or locale ID provides the best processing performance. Using `NULL` will force the function to use the system-wide default locale ID. To return the session parameter that gives the value of the locale of the user running the report, use the following expression: `#$account.parameters.openPagesLocaleId#` Default: `NULL` (without quotes and not case sensitive)
P8	Required. Specifies the delimiter character to be used as a separator for items in selection lists with multiple values (multi-valued enumerations). Typically this value may be a comma (,), semi-colon (;), or pipe (|) character. Default: `','`
P9	Required. Specifies the column on which the selection list values (multi-valued enumerations) will be sorted. The possible values are: • `'EV_DISPLAY_ORDER'` - use to sort the list values by the order of their enumerated value display as they appear in the application user interface. • `'EV_NAME'` - use to sort the list values by the localized enumerated value name. Default: `'EV_DISPLAY_ORDER'`

Table 65. Function Parameters (continued)

Parameter	Description
P10	Required. Specifies the sort direction for items in selection lists with multiple values (multi-valued enumerations). The possible values are: • `'ASC'` - this sorts the list values in ascending order. • `'DESC'` - this sorts the list values in descending order. Default: `'ASC'`

Usage Example

The following example returns a comma-delimited list of selected values for the multi-selection Risk field called Financial Statement Assertions, for each Risk filtered by the selected reporting period. Figure 90 shows sample report output.



Risk Name	Risk Description	Assertions
OPRisk0	OPRisk0 SOXRisk	Completeness, Existence, Occurrence
OPRisk1	OPRisk1 SOXRisk	Accuracy, Obligations

Figure 90. Get MVAL STRAGG Output

Using Time Dependent Functions

This section describes time-based database functions.

Getting the Custom Field Value by ID

This function returns the value for a custom field of an object as of a certain date.

It uses the PropertyDefId to identify the custom field value to return.

Syntax

This function has the following syntax:

```
"OP_RESOURCES_MGR.GET_CUSTOM_FIELD_VALUE_BY_ID" (P1, P2, P3, P4, P5, P6)
```

Where:

Table 66. Function Parameters

Parameter	Description
P1	Required. The object ID (a numeric value) of the object for which the custom field value will be returned.
P2	Required. The as-of-date of the custom field.
P3	Required. The PropertyDefId of the custom field (a numeric value).

Table 66. Function Parameters (continued)

Parameter	Description
P4	Optional. Only use the system audit trail table. • False - a value will be returned for the property regardless of whether it is in the system audit table • True - a value will only be returned if the value is in the system audit trail table. **Note:** When a property is first created, a value is not entered in the system audit trail table. Default: False
P5	`Optional. This field determines the type of output for this function.` The possible values are: • EV_ID • EV_CODE • EV_SYSTEM_NAME • EV_NAME_I18N Default: EV_ID
P6	Optional. The locale ID (a numeric identifier). Used to automatically translate the text of enumerated strings to the locale of the user running the report. **Note:** A value of `NULL` (without quotes) can be used; however, providing the locale session parameter or locale ID provides the best processing performance. Using `NULL` will force the function to use the system-wide default locale ID. To return the session parameter that gives the value of the locale of the user running the report, use the following expression: `#$account.parameters.openPagesLocaleId#` Default: `NULL` (without quotes and not case sensitive)

Note: An error value will be returned if your request precedes the creation of the variable.

Usage Example

The following example shows how to obtain a custom field value by using its PropertyDefId. In this example we want to obtain the Inherent Risk Rating as a function of time. The value 140 is the propertyDefId for the Inherent Risk Rating that was looked up in the propertydefs table using the following SQL:

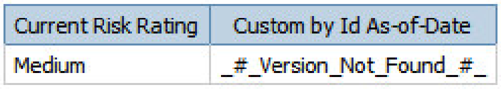
select * from propertydefs where name = 'Inherent Risk Rating'

Once the propertydefId is known, you must decide the format of the output. The example shows the output as EV_NAME_118N which is the name of the property translated to that of the locale ID.

The following is the code we used in this example:

```
"OP_RESOURCES_MGR.GET_CUSTOM_FIELD_VALUE_BY_ID" ([RI_RISK_ID],
to_date(?Date?,'DD-MON-YYYY'), 140, 'N', 'EV_NAME_I18N',
#$account.parameters.openPagesLocaleId#)
```

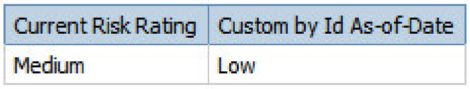
If I select the date 1-Jan-1970 (before the field was created), I obtain the following, shown in Figure 91:



Current Risk Rating	Custom by Id As-of-Date
Medium	_#_Version_Not_Found_#_

Figure 91. Inherent Risk Rating - 1-Jan-1970

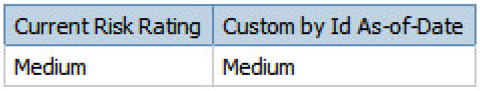
As you can see from the results, the current value of the Risk Rating is Medium, but on the date requested, the value was not found. If I select the date 24-Feb-2011, I get the following, shown in Figure 92:



Current Risk Rating	Custom by Id As-of-Date
Medium	Low

Figure 92. Inherent Risk Rating - 24-Feb-2011

You can now see that on the date 24-Feb-2011, the value of the Inherent Risk Rating was Low. If I enter today's date, I obtain the following, shown in Figure 93:



Current Risk Rating	Custom by Id As-of-Date
Medium	Medium

Figure 93. Inherent Risk Rating - 1-Apr-2011

The value of the field equals the current Inherent Risk Rating.

Getting the Custom Field Value by Name

This function returns the value for a custom property by field group name and property name as a function of date.

Syntax

This function has the following syntax:

```
"OP_RESOURCES_MGR.GET_CUSTOM_FIELD_VALUE_BY_NAME" (P1, P2, P3, P4, P5, P6,
P7)
```

Where:

Table 67. Function Parameters

Parameter	Description
P1	Required. The object ID (a numeric value) of the object for which the custom field value will be returned.
P2	Required. The as-of-date of the custom field.
P3	Required. The name of the field group that contains the custom field.

Table 67. Function Parameters (continued)

Parameter	Description
P4	Required. The name of the custom property.
P5	Optional. Only use the system audit trail table. • False - a value will be returned for the property regardless of whether it is in the system audit table • True - a value will only be returned if the value is in the system audit trail table. **Note:** When a property is first created, a value is not entered in the system audit trail table. Default: False
P6	`Optional. This field determines the type of output for this function.` The possible values are: • EV_ID • EV_CODE • EV_SYSTEM_NAME • EV_NAME_I18N Default: EV_ID
P7	Optional. The locale ID (a numeric identifier). Used to automatically translate the text of enumerated strings to the locale of the user running the report. **Note:** A value of `NULL` (without quotes) can be used; however, providing the locale session parameter or locale ID provides the best processing performance. Using `NULL` will force the function to use the system-wide default locale ID. To return the session parameter that gives the value of the locale of the user running the report, use the following expression: `#$account.parameters.openPagesLocaleId#` Default: `NULL` (without quotes and not case sensitive)

Note: An error value will be returned if your request precedes the creation of the variable.

Usage Example

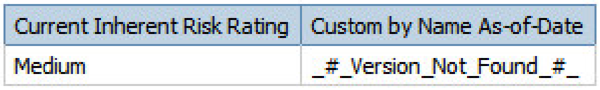
The following example shows how to obtain a custom field value by using its field group and property name. In this example we want to obtain the Inherent Risk Rating as a function of time. We use the UI to look up the field group name for the risk object that contains Inherent Risk Rating.

Once the propertydefId is know, you must decide the format of the output. The example shows the output as EV_NAME_118N which is the name of the property translated to that of the locale ID.

The following is the code we used in this example:

"OP_RESOURCES_MGR.GET_CUSTOM_FIELD_VALUE_BY_NAME" ([RI_RISK_ID], to_date(?Date?,'DD-MON-YYYY'), 'OPSS-Risk-Qual', 'Inherent Risk Rating', 'N', 'EV_NAME_I18N', #$account.parameters.openPagesLocaleId#)

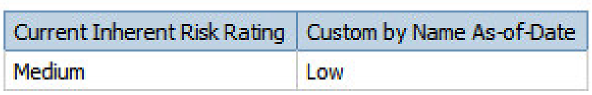
If I select the date 1-Jan-1970 (before the field was created), I obtain the following, shown in Figure 94:



Current Inherent Risk Rating	Custom by Name As-of-Date
Medium	_#_Version_Not_Found_#_

Figure 94. Inherent Risk Rating - 1-Jan-1970

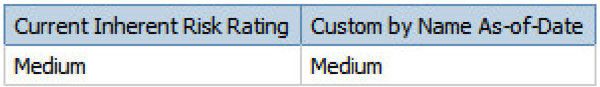
As you can see from the results, the current value of the Risk Rating is Medium, but on the date requested, the value was not found. If I select the date 24-Feb-2011, I get the following, shown in Figure 95:



Current Inherent Risk Rating	Custom by Name As-of-Date
Medium	Low

Figure 95. Inherent Risk Rating - 24-Feb-2011

You can now see that on the date 24-Feb-2011, the value of the Inherent Risk Rating was Low. If I enter today's date, I obtain the following, shown in Figure 96:



Current Inherent Risk Rating	Custom by Name As-of-Date
Medium	Medium

Figure 96. Inherent Risk Rating - 1-Apr-2011

The value of the field equals the current Inherent Risk Rating.

Getting the System Field Description

This function returns the value for the description property of an object as a function of time.

Syntax

This function has the following syntax:

```
"OP_RESOURCES_MGR.GET_SYSTEM_FIELD_VALUE_DESC" (P1, P2, P3)
```

Where:

Table 68. Function Parameters

Parameter	Description
P1	Required. The object ID (a numeric value) of the object for which the custom field value will be returned.
P2	Required. The as-of-date of the system field.

Table 68. Function Parameters (continued)

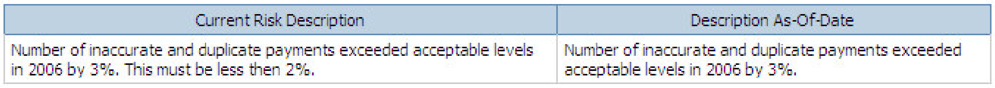
Parameter	Description
P3	Optional. Only use the system audit trail table. • False - a value will be returned for the property regardless of whether it is in the system audit table • True - a value will only be returned if the value is in the system audit trail table. **Note:** When a property is first created, a value is not entered in the system audit trail table. Default: False

Note: An error value will be returned if your request precedes the creation of the variable.

Usage Example 1

The following example shows the use of this function call to obtain the system description property as a function of time. The following expression was used:

In the first example we obtained the system description value as of 1-Feb-2011. This is after the object was created, but before it was modified. The Current® Risk Description field always shows the value of the field as it exists when the report is run, shown in Figure 97.

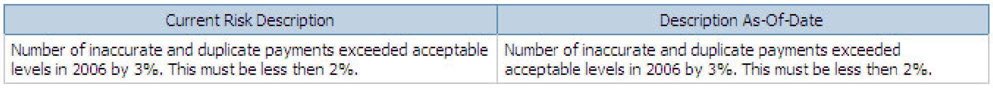


Current Risk Description	Description As-Of-Date
Number of inaccurate and duplicate payments exceeded acceptable levels in 2006 by 3%. This must be less then 2%.	Number of inaccurate and duplicate payments exceeded acceptable levels in 2006 by 3%.

Figure 97. Get System Description Value - February 1, 2011

Usage Example 2

In the next example, shown in Figure 98, we obtained the system description value as of 11-Feb-2011. Notice the Description As-of-Date now equals the Current Risk Description value. This is after the final modification of the object.

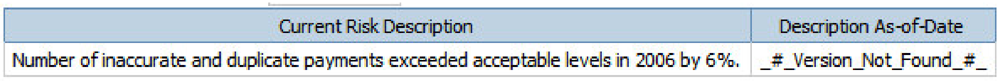


Current Risk Description	Description As-Of-Date
Number of inaccurate and duplicate payments exceeded acceptable levels in 2006 by 3%. This must be less then 2%.	Number of inaccurate and duplicate payments exceeded acceptable levels in 2006 by 3%. This must be less then 2%.

Figure 98. Get System Description Value - February 11, 2011

Usage Example 3

In the final example, shown in Figure 99, we obtained the system description value as of 11-Feb-2009, before it existed.



Current Risk Description	Description As-of-Date
Number of inaccurate and duplicate payments exceeded acceptable levels in 2006 by 6%.	_#_Version_Not_Found_#_

Figure 99. Get System Description Value - February 11, 2009

We see the system returns an error value that the property was not found.

Getting the System Field Full Path

This function returns the value for the full path property of an object as a function of time.

Syntax

This function has the syntax "`OP_RESOURCES_MGR.GET_SYSTEM_FIELD_VALUE_FULLPTH`" `(P1, P2, P3)` where the following table describes the variables.

Table 69. Function Parameters

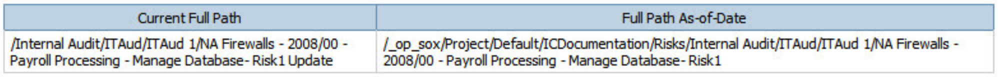
Parameter	Description
P1	Required. The object ID (a numeric value) of the object for which the custom field value will be returned.
P2	Required. The as-of-date of the system field.
P3	Optional. Only use the system audit trail table. • False - a value will be returned for the property regardless of whether it is in the system audit table • True - a value will only be returned if the value is in the system audit trail table. **Note:** When a property is first created, a value is not entered in the system audit trail table. Default: False

Note: An error value will be returned if your request precedes the creation of the variable.

Usage Example 1

The following example shows the use of this function call to obtain the system full path property as a function of time. The following expression was used:

In the first example, shown in Figure 100, we obtained the system full path value as of 1-Feb-2011. This is after the object was created, but before it was modified.

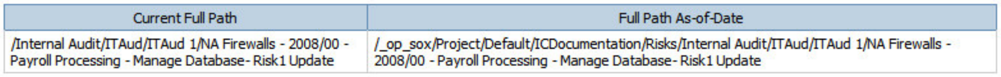


Current Full Path	Full Path As-of-Date
/Internal Audit/ITAud/ITAud 1/NA Firewalls - 2008/00 - Payroll Processing - Manage Database- Risk1 Update	/_op_sox/Project/Default/ICDocumentation/Risks/Internal Audit/ITAud/ITAud 1/NA Firewalls - 2008/00 - Payroll Processing - Manage Database- Risk1

Figure 100. Get System Full Path Value - Example 1

Usage Example 2

In the next example, shown in Figure 101 on page 192, we obtained the system full path value as of 11-Feb-2011. Notice the Description As-of-Date now equals the current full path value. This is after the final modification of the full path.

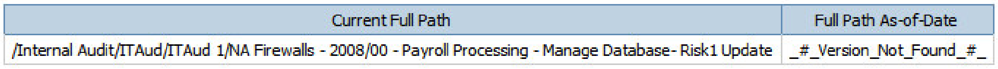


Current Full Path	Full Path As-of-Date
/Internal Audit/ITAud/ITAud 1/NA Firewalls - 2008/00 - Payroll Processing - Manage Database- Risk1 Update	/_op_sox/Project/Default/ICDocumentation/Risks/Internal Audit/ITAud/ITAud 1/NA Firewalls - 2008/00 - Payroll Processing - Manage Database- Risk1 Update

Figure 101. Get System Full Path Value - Example 2

Usage Example 3

In the final example, shown in Figure 102, we obtained the system full path value as of 11-Feb-2009, before it existed.



Current Full Path	Full Path As-of-Date
/Internal Audit/ITAud/ITAud 1/NA Firewalls - 2008/00 - Payroll Processing - Manage Database- Risk1 Update	_#_Version_Not_Found_#_

Figure 102. Get System Full Path Value - Example 3

We see the system returns an error value that the property was not found.

Getting the System Field Name

This function returns the value for the name property of an object as a function of time.

Syntax

This function has the following syntax:

```
"OP_RESOURCES_MGR.GET_SYSTEM_FIELD_VALUE_NAME" (P1, P2, P3)
```

Where:

Table 70. Function Parameters

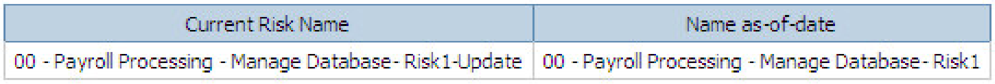
Parameter	Description
P1	Required. The object ID (a numeric value) of the object for which the custom field value will be returned.
P2	Required. The as-of-date of the system field.
P3	Optional. Only use the system audit trail table. • False - a value will be returned for the property regardless of whether it is in the system audit table • True - a value will only be returned if the value is in the system audit trail table. **Note:** When a property is first created, a value is not entered in the system audit trail table. Default: False

Note: An error value will be returned if your request precedes the creation of the variable.

Usage Example 1

The following example shows the use of this function call to obtain the system name property as a function of time. The following expression was used:

In the first example, shown in Figure 103, we obtained the system name value as of 1-Feb-2011. This is after the name was created, but before it was modified.

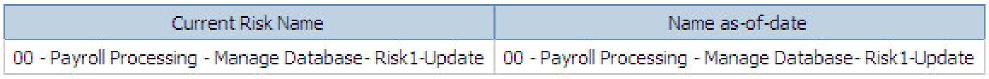


Current Risk Name	Name as-of-date
00 - Payroll Processing - Manage Database- Risk1-Update	00 - Payroll Processing - Manage Database- Risk1

Figure 103. Get System Name Value - February 1, 2011

Usage Example 2

In the next example, shown in Figure 104, we obtained the system name value as of 11-Feb-2011. Notice the Name as-of-date now equals the Current Risk Name. This is after the final modification of the name.

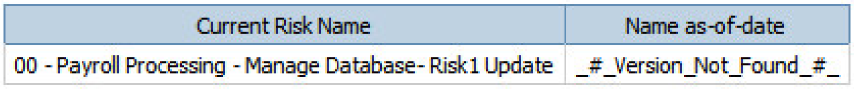


Current Risk Name	Name as-of-date
00 - Payroll Processing - Manage Database- Risk1-Update	00 - Payroll Processing - Manage Database- Risk1-Update

Figure 104. Get System Name Value - February 11, 2011

Usage Example 3

In the final example, shown in Figure 105, we obtained the system name value as of 11-Feb-2009, before it existed.



Current Risk Name	Name as-of-date
00 - Payroll Processing - Manage Database- Risk1 Update	_#_Version_Not_Found_#_

Figure 105. Get System Name Value - February 11, 2009

You can see that the system name is not found.

Getting the System Field Parent Folder ID

This function returns the value for the parent folder ID property of an object as a function of time.

Syntax

This function has the following syntax:

```
"OP_RESOURCES_MGR.GET_SYSTEM_FIELD_VALUE_PFLDRID" (P1, P2, P3)
```

Where:

Table 71. Function Parameters

Parameter	Description
P1	Required. The object ID (a numeric value) of the object for which the custom field value will be returned.
P2	Required. The as-of-date of the system field.

Table 71. Function Parameters (continued)

Parameter	Description
P3	Optional. Only use the system audit trail table. • False - a value will be returned for the property regardless of whether it is in the system audit table • True - a value will only be returned if the value is in the system audit trail table. **Note:** When a property is first created, a value is not entered in the system audit trail table. Default: False

Note: An error value will be returned if your request precedes the creation of the variable.

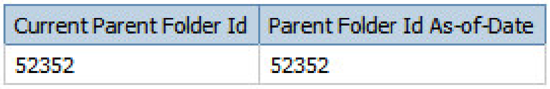
Usage Example 1

The following example shows the use of this function call to obtain the system parent folder ID property as a function of time. The following expression was used:

```
"OP_RESOURCES_MGR.GET_SYSTEM_FIELD_VALUE_PFLDRID" ([RI_RISK_ID],
  to_date(?Date?,'DD-MON-YYYY'))
```

In the first example we obtained the system parent folder ID as of 1-Feb-2011. This is after the object was created, but before it was modified in any manner. We can see in Figure 106 that modifying an attribute on an object does not affect the parent folder ID.

Note: Self-contained object types do modify the folder if you change the name of the object.

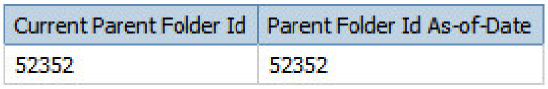


Current Parent Folder Id	Parent Folder Id As-of-Date
52352	52352

Figure 106. Get System Parent Folder ID - February 1, 2011

Usage Example 2

In the next example, shown in Figure 107, we obtained the system parent folder ID as of 11-Feb-2011. Notice that once again, this value has not changed. This is after the final modification of the name.

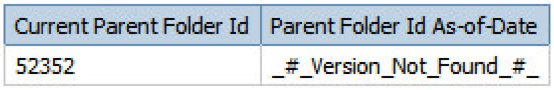


Current Parent Folder Id	Parent Folder Id As-of-Date
52352	52352

Figure 107. Get System Parent Folder ID - February 11, 2011

Usage Example 3

In the final example, shown in Figure 108, we obtained the system parent folder ID as of 11-Feb-2009, before it existed.



Current Parent Folder Id	Parent Folder Id As-of-Date
52352	_#_Version_Not_Found_#_

Figure 108. Get System Parent Folder ID - February 11, 2009

We see the system returns an error value that the property was not found.

Chapter 14. Formatting Reports for Output

Besides the standard HTML output, the three most popular output formats are PDF, Excel and CSV. In order to be able to use the formats effectively, this chapter describes some best practices for formatting your reports so they will appear well formatted when output.

Setting Up the Report for PDF Output

Follow this procedure to set up your report for PDF output.

Procedure

1. Click on **File | PDF Page Setup...**.
2. On the PDF setup page select portrait or landscape mode depending on the size and style of your report. Leave the page size as default so as not to force the printer to have a certain size paper.

Including In-Line Prompts for PDF Output

Follow this procedure to see a business entity prompt in PDF output.

In-line prompts will not appear in a PDF file. When you convert a report to a PDF, you will notice that the prompt and the button have been stripped out of the format.

Procedure

1. Place a second header line in your report with only the prompt name and value on the line.
2. Create a string condition using the **Conditional Explorer** with the following code:

   ```
   if (ReportOutput() starts with 'XL')
   then ('XLS')
   else (ReportOutput())
   ```
3. Rename the variable to **Report Output**.
4. Add as values, XLS, PDF, HTML.
5. On the **table row** with the in-line prompt, set the **Style Variable** to **Report Output**.
6. From the **Conditional Explorer**, select **PDF** under **Report Output**.
7. In the **Properties Pane**, under **Box**, set the **Box Type** to None.
8. Now repeat the same steps for the row without the in-line prompt but select HTML instead of PDF.
9. Run the report in HTML.

Results

The report looks the same as you had before in HTML.

Now when you convert it to a PDF, you see prompt row you just created.

Formatting the Report for PDF Output

Formatting the report for PDF output or for HTML output share many of the same formatting techniques. Any of the formatting techniques used to adjust the layout in an HTML page can be used to adjust the format of the PDF file. Just remember that if you are using both HTML and PDF outputs, you must sometimes make a trade off in terms of which format to optimize.

If we start with a simple report which shows a hyperlinked entity name and the full path of the same entity, we can make changes to the report and show how they are reflected in the PDF output.

Procedure

1. Set the font of the full path variable to Bold.
2. Set the width of the name variable to 300 px.
3. Set the border to 1 1/2 pt, and red.
4. Run the report.

Results

The basic formatting techniques you use for HTML also apply to the PDF format.

Formatting for Microsoft Excel Output

When designing a report to be formatted for Microsoft Excel spreadsheet software, the key limitation is that all containers in the report must have the same number of columns.

Microsoft Excel spreadsheets do not depend on column width values like a PDF or HTML formatted report, it strictly relies on column quantity to align one container to the other.

What this means is that, headers, footers, lists and tables must have dummy columns added to them to cause them to line up in the output. These columns don't require data, they just need to exist.

By adding the extra columns, the alignment of the columns in the Excel report improves.

Formatting for CSV Output

The CSV format (comma separated values) is the most basic output format in that there is no style formatting.

This type of output is designed to be outputted in a manner that can be read by other systems. The only known limitation of this format is that you cannot have joins in the query or IBM Cognos will produce an error. To avoid this problem you can create a single CQS which you use for the CSV output.

When the report is generated, it will still appear in Excel format and the **File Download** dialog will open. This is similar to the Excel dialog with no indication it is for a CSV file.

When you open the file you will see the file is correctly formatted for CSV (Showing two columns, with the second column clipped due to its size). In CSV formatting you can't have word wrap.

To save this in its final format, click on **Save As** and select a file name and the file type as **CSV**.

When you open the .txt file in Notepad, you will see the commas separating the two columns and when there is a comma itself in the data, you can also see the double quote at the beginning of the second column for some records indicating that this problem is resolved.

Chapter 15. Authoring Reports

This section describes the techniques used to create reports using the IBM OpenPages GRC on Cloud framework. It is a detailed example of the tasks used to develop reports and when to apply them.

Manual IBM Cognos Configuration Settings

There are numerous IBM Cognos environment settings in IBM OpenPages GRC on Cloud. Some of these have been set by the OpenPages GRC on Cloud installer while others have to be manually set. This section will describe those that need to be set manually.

Drill Behavior

When creating a dimensional report where drill-up/drill-down is required, there is a setting which must be checked in order for this behavior to function.

From the menu bar, select **Data | Drill Behavior**, the **Drill Behavior** dialog opens.

In order for drill-up/drill-down to function, **Allow drill-up and drill-down** must be checked.

Validation Options

By default IBM Cognos will show warning messages when you validate a formula or the entire report. You have the option of suppressing these. In order to do this, go to the menu bar and select **Tools | Validate Options...**.

Set the **Validation level:** to **Error** and the warning messages will no longer appear.

Default Data Formats

If you are developing reports that require data styled in certain formats, you might want to set the default data formats in your project report template. In order to set these, go to the menu bar and select **Data | Default Data Formats...**.

Set the default properties of data item types so when you insert an item of that type into the report it will inherit these properties.

How to Configure Reports

There are numerous items that need to be configured in IBM OpenPages GRC on Cloud before the report author can begin. Some of the items that need to be considered are as follows:

- Namespaces - You will have been supplied OOB namespaces that support the reports delivered by OpenPages GRC on Cloud. These may or may not support the needs of your reporting. You have the ability to extend these namespaces or create completely new namespaces to meet your needs. Namespaces are global objects.
- Fact Types - The types of summary functions that will be needed in your dimensional model. This is a global setting that affects all namespaces.

- Locales - Which languages you want your framework to support. This is a global setting that affects all namespaces.
- Triangles - Whether your report will need to utilize a triangle due to a recursive object. This setting is defined at both the global and namespace level.
- Auxiliary Tables - Whether you need to pull tables/CQS into a namespace to support your reports. This is defined at the namespace level.
- Object Prefixes - If custom objects have been added to the system, the object prefix has to defined. These are global settings.
- Facts and Dimensions - If you are utilizing the dimensional model, you will need to define facts and dimensions for the objects you will be reporting from. This is defined at the object level.
- Business Entity Recursive Object Levels - These are the user defined business entity hierarchies that can be used to create a legal vs. business reporting structure for example.
- Entity Recursive Objects Levels - Which Business Entity Recursive Object Levels you want in each namespace.
- Object Type Dimensions - These are the hierarchies that can be defined to be used in the dimensional namespace. You will need at least one of these to create useful dimensional reporting. The most common Object Type Dimension is a business entity dimension. This is a global setting.
- "All" Dimension Text - This setting determines what is displayed when you drill-up to the "All" level in a hierarchy. This is a global setting that affects all dimensions.
- Index Creation - Depending on the design of the report, you might need to add an index to a field to improve performance. This will only affect the table you add the index to.

Besides the above configurable items, there are also customer field definitions that must be loaded in order to start report development. These include the following:

- Field Bundles/Groups - These are collections of fields that are assigned to an object.
- Fields - This is the definition of an individual field that the customer has asked for to facilitate the configuration of their system.
- Enumerate String Values - These are the list of allowable values for enumerated strings.

All of the above items require that the framework be regenerated once a change is made to them. The goal here is to investigate what needs to be defined, make the configuration changes and then generate your framework.

Underlying Report SQL

There is a key choice to be made on which type of underlying construct is used to create the report. You must decide the following:

- Using Namespaces - With the advent of the V6 framework, the generated SQL efficiency has been significantly improved over that of the legacy framework. The current best practice is to use this framework and add namespaces to create your desired reports.
- Using Custom Query Subjects (CQS) - CQS files need to be used when the underlying SQL that needs to be generated does not follow a top to bottom approach or it requires joins between objects that are non-standard. The decision to use a CQS for a report is no longer just based on the expected performance of

the report. The V6 model generates equal or better performing SQL then can be created by hand. The decision is based upon whether the framework can support the underlying SQL construct to create the report.

We are going to assume that the report to be created will use namespaces to create the underlying SQL. Creating Custom Query Subjects is explained in Chapter 18, “Custom Query Subjects,” on page 219.

Using a Report Template

The best place to start is with the report template that should have been developed during the report requirements phase of the project. The template reflects the style in which all reports for this project should be developed too. This template includes useful conditional formulas, the layout for the prompt page, the layout for the report page and pre-defined query subjects for business entities and reporting periods. By starting with a template you are more likely to create uniform reports that can be edited by any report author in your organization.

- The prompt page contains the two most popular prompts and their underlying SQL. These prompts are obtained from the DEFAULT namespace, if another namespace is used in this report, you want to adjust the origin of these data fields. In addition, the default spacing of the table cells has been set.
- The report page has the header configured with the logo, the javascript for hyperlinks loaded, the layout of sub-headers defined, a dummy list has been added and the footer setup.
- The logo is located on the page as an example for the customer. It can go on either the right or left hand sides. In order to replace the logo, you must replace the logo.gif file with the correct logo on the IBM Cognos server.
- The html object contains the javascript for the hyperlinks. If you don’t use hyperlinks in this report, you can remove this object. The dummy list contains the display portion of the hyperlink code as an expression defined on the hyperlink object.
- The header is setup to be "squeezable" so that when you want to make the report less than a full screen, the titles will squeeze properly.
- The no data displayed value has been set up on the dummy list object. This appears when the user selects a combination of prompts that yields no data in the report.
- Lastly, the footer has been configured to be localized. The date and time formats are set up to be generic so that IBM Cognos will localize these fields depending on the locale associated with the user’s login.
- Common conditional formulas have been included in the template:

  The following describes the purpose of the conditional formulas:
  - Localization - This formula is used to localize text fields to the supported languages besides U.S. English.
  - Report Header - Since there is no concept of a report header in IBM Cognos, this formula simulates one with the page header.
  - Row Highlight - This is used to highlight every other row in a list report. This is only useful for simple list reports.
  - Report Output - This is used when you need to create a condition depending on the report output type.

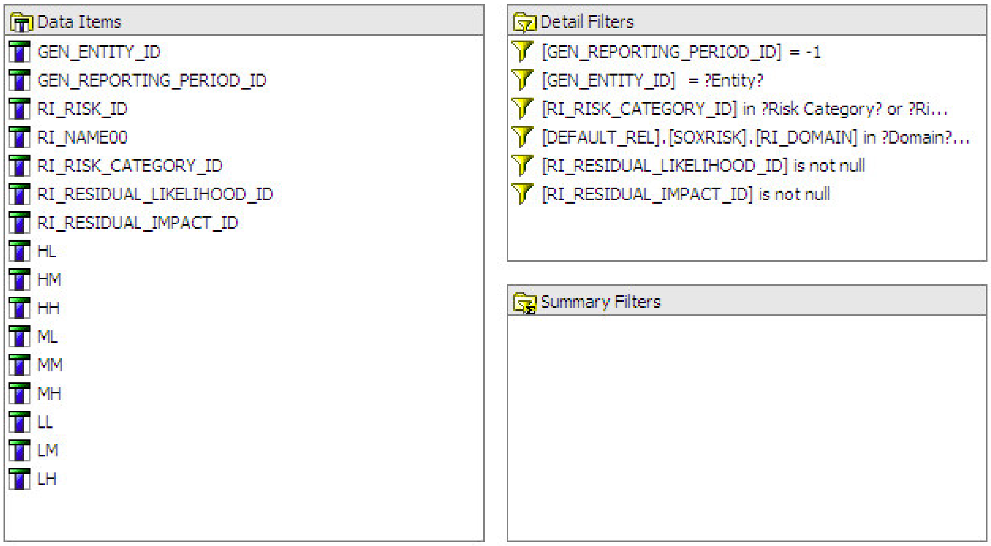
Note: Contact you project lead to obtain a copy of the IBM OpenPages GRC on Cloud template.

Query Subject Organization

When creating query subjects for reports the following guidelines should be observed:

- Place the prompt queries at the bottom of the page. They are used infrequently and you don't want to constantly scroll through them.
- Place supporting queries for a join or union directly under that construct. It makes them easier to find.
- Name your query subjects in a logical manner so the next report author can understand the report.

Typically the query subjects are designed from the bottom up. The most detailed queries at the bottom and the most aggregated queries at the top, shown in Figure 109. The idea is to populate query subjects with the values that are required to meet the requirements of the report. Constructs such as filters are placed on the lowest level query subject in the hierarchy in order to speed up performance.



GEN_ENTITY_ID
GEN_REPORTING_PERIOD_ID
RI_RISK_ID
RI_NAME00
RI_RISK_CATEGORY_ID
RI_RESIDUAL_LIKELIHOOD_ID
RI_RESIDUAL_IMPACT_ID
HL
HM
HH
ML
MM
MH
LL
LM
LH

[GEN_REPORTING_PERIOD_ID] = -1
[GEN_ENTITY_ID] = ?Entity?
[RI_RISK_CATEGORY_ID] in ?Risk Category? or ?Ri...
[DEFAULT_REL].[SOXRISK].[RI_DOMAIN] in ?Domain?...
[RI_RESIDUAL_LIKELIHOOD_ID] is not null
[RI_RESIDUAL_IMPACT_ID] is not null

Figure 109. Query Subject Children

What can be seen in the above example are as following:

- There are fields included in the report from the business entity object and the risk object.
- There are custom data items added to this query.
- There are filters added to this query because it is the child query of the report.

As you build each query subject you should be checking the data that is being produced to make sure the correct data is pulled from the database and duplicate records are not included. By doing this at every query subject you increase the quality of what you are building and your understanding of how the data is constructed. You must be aware that IBM OpenPages GRC on Cloud is a very flexible system that allows an unlimited amount of configuration. If your understanding of the data model differs from the customer's, you will produce the wrong results.

You can produce output from any of the query subjects by selecting an item in the query subject and using the **View Tabular Data**. This will bring up the prompts that are included in the query subject (and any query subject that is a child of the query subject you are displaying). To perform this action select the following from the **Run** menu.

A prompt page will appear that will need to be filled in with valid values and the output of the query subject will be displayed in a basic list format. This displays all the data the query returned from the database and you can now review it to determine if this is the data you expected.

Query Subject Constructs

Due to the existence of joins in IBM Cognos, there are fewer query subjects to select from than in the legacy framework. All standalone objects must be manually joined along with any primary objects that need to be joined due to the complexity of a report.

In addition, some of the constructs have been updated. For instance unions can now handle more than two query subjects. This creates more efficient SQL than in previous versions of IBM Cognos.

The union now also includes the following properties:

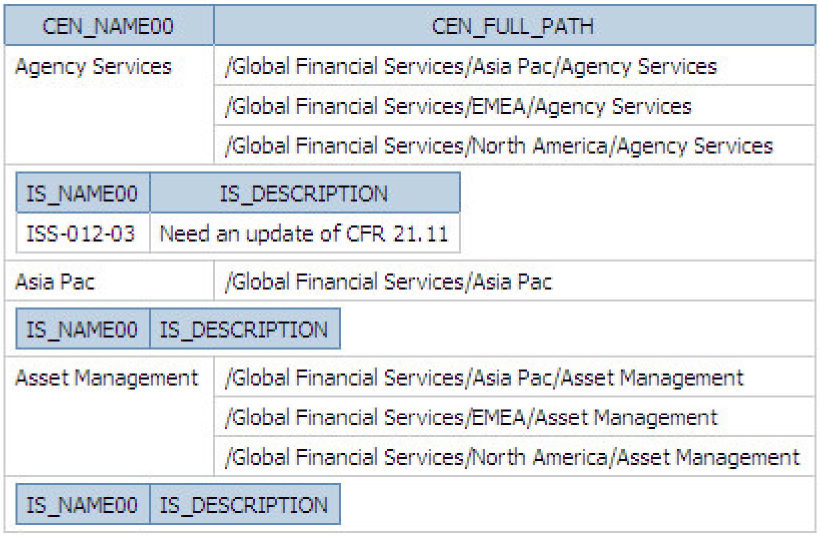
- Set Operation - This enables the user to change the operation to another type of set operator. This should always be set to **Union**.
- Duplicates - This determines whether duplicate rows are removed from the result set. This is normally set to **Remove**.
- Projection List - This determines whether the set of data items that is a result of this union are automatically or manually generated. We recommend that this is always set to **Automatically Generated**. This is due to issues we have seen when not all the data set items were included in the result set (when the selection was manually) and it caused validation problems.

Formatting Empty Tables or Lists

An issue that must be addressed is that of blank lists/tables/crosstabs in a report. If the primary container returns no data, then using the **No Data Contents** property of the container will inform the user that they have chosen prompt values that yield no data.

The second situation is when you have a sub-table or list in a report and it has no data in it when the report is run. What you see is a header and nothing else.

In the following example we have a report that displays entity information and in the list footer we display a list of issues attached to that entity. When we first run the report we see the following, shown in Figure 110 on page 206:



CEN_NAME00	CEN_FULL_PATH
Agency Services	/Global Financial Services/Asia Pac/Agency Services
	/Global Financial Services/EMEA/Agency Services
	/Global Financial Services/North America/Agency Services
IS_NAME00	IS_DESCRIPTION
ISS-012-03	Need an update of CFR 21.11
Asia Pac	/Global Financial Services/Asia Pac
IS_NAME00	IS_DESCRIPTION
Asset Management	/Global Financial Services/Asia Pac/Asset Management
	/Global Financial Services/EMEA/Asset Management
	/Global Financial Services/North America/Asset Management
IS_NAME00	IS_DESCRIPTION

Figure 110. Blank Section Shown

What we see is that in the first section there is an issue listed while in the 2nd and 3rd sections there is no data found and the header of the list appears.

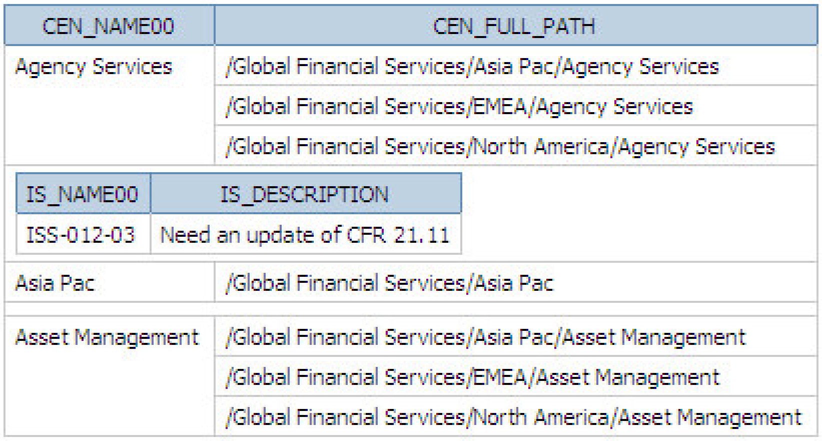
In order to remove these blank sections, do the following:

Procedure

1. Using the **Condition Explorer**, do the following:
   a. Create a new Boolean Variable Boolean variable.
   b. In the Boolean variable, enter the following formula;
   ```
   [Issue].[IS_ISSUE_ID] is missing or
   [Issue].[IS_ISSUE_ID] is null
   ```
   c. Click **OK**.
   d. In the Properties pane, under **Miscellaneous**, set the **Name** property to **Hide Titles**.
2. In the main report page, do the following:
   a. Select a list header in the issue list.
   b. Using the ancestor selector, select **List Columns Title Style.**
   c. In the Properties pane, under the **Conditional** section, set the **Style Variable** to **Hide Titles**.
3. Now in the **Condition Explorer**, do the following:
   a. Select **Yes** under the **Hide Titles** variable.
   b. In the Properties pane, in the **Box** section, set the **Box Type** to **None**.
4. Run the report.

Results

What you now see, shown in Figure 111 on page 207, when you run the report is that the list header title has been removed when there are no data items in the list.



<table>
<tr><th>CEN_NAME00</th><th>CEN_FULL_PATH</th></tr>
<tr><td>Agency Services</td><td>/Global Financial Services/Asia Pac/Agency Services</td></tr>
<tr><td></td><td>/Global Financial Services/EMEA/Agency Services</td></tr>
<tr><td></td><td>/Global Financial Services/North America/Agency Services</td></tr>
<tr><td colspan="2"><table><tr><th>IS_NAME00</th><th>IS_DESCRIPTION</th></tr><tr><td>ISS-012-03</td><td>Need an update of CFR 21.11</td></tr></table></td></tr>
<tr><td>Asia Pac</td><td>/Global Financial Services/Asia Pac</td></tr>
<tr><td colspan="2"></td></tr>
<tr><td>Asset Management</td><td>/Global Financial Services/Asia Pac/Asset Management</td></tr>
<tr><td></td><td>/Global Financial Services/EMEA/Asset Management</td></tr>
<tr><td></td><td>/Global Financial Services/North America/Asset Management</td></tr>
</table>

Figure 111. Hiding Blank List

Chapter 16. Computed Fields

Computed fields bring the ability to define fields on an object whose value is computed from the values of other fields. These other fields may exist on the same object or another related object. These computations will be available in both the application as well as for reporting purposes.

Computed fields are defined in the application user interface. For information about defining a new computed field, see the *IBM OpenPages GRC on Cloud Administrator's Guide* or contact your system administrator for assistance.

Limitations of Computed Fields

- Currently there is a limitation on the size of the computation attribute value that can be stored by the application. The main attribute of concern is 'Equation' where a complex flattened equation could be very lengthy.

  There is a 20,000 byte limit on the size of the entered text. Note that the application supports multi-byte characters and so this may not be the equivalent of 20,000 characters.

  If more than 20,000 bytes of information are needed for the equation you must create the calculation in the custom query subject file as described in "Creating Calculations" on page 228.
- When exploring all the computation possibilities there is one large distinction in what should be done. While it is possible to query up the relationship tree (that is, compute values based on ancestors) it is strongly discouraged.

  The automatic framework generation is set up in such a way as to create joins that are conducive to better performance querying down the relationship tree.

  Note: A query up the tree will result in bad computed field performance as well as place a large strain on the database that can result in the entire application slowing down.
- Computed fields can sometimes act as building blocks for other computed fields. These are referred to as intermediate computations. Currently the IBM OpenPages GRC on Cloud application does not support intermediate calculation definitions through the application user interface. If you want to reference another computed field, you must replicate the equation used in that computed field inside the equation for the current field.

  Example

  For example, if we have a computed field `A` and define it as `A = B * C` and we also know `C = D + E`, we would only create one computed field `A` in the application where the equation would be `B * (D + E)`. While this approach can be verbose, it is sometimes the simplest.
- The definitions used in a computed field must be present in both the namespace you are using and be related to the object you are creating the computed field within. It is possible to create a computed field that will validate in the form but will not yield any results due to this limitation.

Design Considerations

Take the following into account when designing the computed field:

- Localization – All computed data types will be localized by the application except for "Simple String" which must be localized as part of the computation (equation).
- Dependent Computed Fields – A computation that uses another computation as part of its equation is not supported.
- Computation Cross Products – When writing a computation that aggregates across multiple object types we commonly run into a problem where some objects are counted multiple times and so you need to account for this.
- Handling Null Values - When writing an equation you must take into consideration that if part of the equation evaluates to null and is then used in some other arithmetic operation, the second operation will return null as well. The solution to this problem is the `nvl()` function which takes two arguments: an expression to evaluate and check if it is null and a value to use in the case it is null. For example:

  ```
  average (nvl(Loss Impacts, 0) for Loss Events) * count
         ( nvl(distinct Loss Impacts for Loss Events, 0))
  ```

Creating a Computed Field

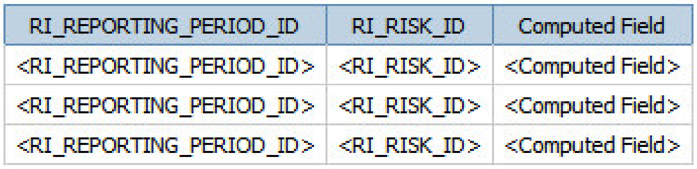
Follow this procedure to create a computed field.

Procedure

1. Create a new list report.
2. Add the following query items to the list:
   - `DEFAULT|DEFAULT_REL|GRC_OBJECTS|[SOXRISK]|ID_FIELDS|[RI_REPORTING_PERIOD_ID]`
   - `DEFAULT|DEFAULT_REL|GRC_OBJECTS|[SOXRISK]|ID_FIELDS|[RI_RISK_ID]`
3. Click the **Toolbox** tab in the **Insertable Objects** pane.
4. Drag and drop a Query Calculation **Query Calculation** object onto the **list**.
5. Name the Query Calculation **Computed Field**.
6. In the Expression Definition pane enter the following expression:

   ```
   total([DEFAULT_REL].[SOXRISK].[RI_INHERENT_SEVERIT_BA] for
    [DEFAULT_REL].[SOXRISK].[RI_RISK_ID]) / 100
   ```

7. ☑ Validate the expression, and then click **OK**.
8. Your report should now look as follows:



RI_REPORTING_PERIOD_ID	RI_RISK_ID	Computed Field
<RI_REPORTING_PERIOD_ID>	<RI_RISK_ID>	<Computed Field>
<RI_REPORTING_PERIOD_ID>	<RI_RISK_ID>	<Computed Field>
<RI_REPORTING_PERIOD_ID>	<RI_RISK_ID>	<Computed Field>

Figure 112. Computed Field Added to a List Report

9. Run the report.

Supplying Required Fields

In order for the system administrator to create the computed field in IBM OpenPages GRC on Cloud, you must supply the following fields: Equation, Primary Namespace, Alternate Namespaces, Object ID Column, and Reporting Period ID Column.

- **Equation**: The equation is the SQL used to define the computed value for the object field. It can be a reference to an existing query item in the published framework or an equation involving multiple query items.
- **Primary Namespace**: The primary namespace is the framework namespace in which the computation is to be performed.

  Note: All referenced query items in the values for Equation, Object ID Column, and Reporting Period ID Column must be in the same namespace.
- **Alternate Namespaces**: The alternate namespace is another framework namespace which the computation will be added during reporting framework generation.
- **Object ID Column**: The Object ID Column is a reference to a framework query item that contains the resource ID of the computed field's object type. This value must be the same for all computed fields in a given namespace for an object type.
- **Reporting Period ID Column**: The Reporting Period ID Column is the framework query item that contains the Reporting Period ID of the computed field's object type. This value must be the same for all computed fields in a given namespace for an object type.

  Note: The Resource ID and Reporting Period ID must match within the field group and object type. If these values do not match, the validation will fail.

Creating a Data File

To obtain the required field values, the best method is to cut and paste the values from into a text file and send them to your System Administrator so the field can be created. Follow this procedure to obtain the values.

Procedure

1. At the top of the file create a label called **Name** and give the computed field a name.
2. Create a label called **Description** and enter a value for the description of the computed field.
3. Double-click on the Computed Field column.
4. Cut and paste the equation from the Expression Definition pane into a text file. Make sure to label it as **Expression** in the file.
5. Click **Cancel**.
6. Create a label for the **Primary Namespace** and enter the name of the namespace into the text file. This is top level namespace NOT the sub-namespace.
7. If there are **Alternate Namespaces** that will use this calculation, create the appropriate element in the text file.
8. Repeat steps 3-5 for the **Object ID** and **Reporting Period** fields labeling them appropriately in the text file.

Results

You should now send the text file to your System Administrator and they can add the computed field to the system. The instructions for doing this are in the *IBM OpenPages GRC on Cloud Administrator's Guide.*

Using Computed Fields in Multiple Namespaces

In the IBM OpenPages GRC on Cloud application we allow multiple parent object types for a given child object type.

This is not true in IBM Cognos for a given namespace due to some of the technical limitations in IBM Cognos. For example, in the ORM namespace the only path to a Loss Event is through a Business Entity. This means that if a Loss Event is associated to a Risk, but not a Business Entity, that Loss Event will not show up in queries against that namespace. We work around this problem by introducing multiple namespaces where each namespace has a distinct path to an object type.

In order to make the calculation available in multiple namespaces for report authors, we have the Additional Namespaces field. This is a comma-delimited list of alternate namespaces for which a Calculation object should be created during the framework generation process. During this process, a calculation object is first created for the primary namespace using the value from the Equation attribute. Then it creates other calculation objects in other namespaces by taking the equation and substituting the alternate namespaces for the primary namespace.

Important: An equation may be valid in one namespace, it may be invalid in others. While in most cases this is not a problem, if the query subject name or query item name varies across namespaces you may need to create separate computed field instances with different equations.

Chapter 17. Reporting Fragments

A reporting fragment is typically either a chart, crosstab or list that can be placed on a page of the application so that the user can see a visual representation of data alongside the page data.

Limitations on Reporting Fragments

The following are the known limitations when implementing reporting fragments:

- Page breaks are not supported.
- Tooltips are not supported.
- A report that has required prompts other than Object ID and Reporting Period ID cannot be used.
- In-line prompts on the report page are not supported.
- You cannot set the background color of a fragment to transparent.
- The method for implementing CrossTracks is different in a reporting fragment.
- There can only be one reporting fragment in a report. Placing multiple reporting fragments in a single report will cause all of the fragments to run every time a single fragment is run in the UI.
- The fragment can only be used with the object that is being passed to the report as a prompt.
- Drill-throughs are not supported.
- The "No Data Contents" property does not work for data containers.

Creating Reporting Fragments

Reporting fragments are developed in the same manner as that of a report. A reporting fragments is a single data container (most likely a chart or a list) with data. There are no headers, footers, or other report adornments.

The report fragment, when rendered in the application, is either a picture (in the case of a chart) or an HTML table in the case of a list or a crosstab. The only interactive feature that is available to you is that of a hyperlink which is described in the next section.

There are a few requirements when creating your fragment:

- You must pass a Resource ID to the fragment. In your report, this will take the form of a filter:

  ```
  [DEFAULT_REL].[SOXBUSENTITY_GPC].[GEN_ENTITY_ID] = ?Business Entity?
  ```

- You should also pass the Reporting Period ID to the report. Once again it will take the form of a filter:

  ```
  [DEFAULT_REL].[SOXBUSENTITY_GPC].[GEN_REPORTING_PERIOD_ID] =
  ?Reporting Period ID?
  ```

- You will have to give the report element that you want to publish as a fragment a name:
  1. Select the data container (chart, list or crosstab) and using the ancestor selector in the **Properties** pane, select the container itself.
  2. Under the **Miscellaneous** category, set the **Name** property to listMain or whatever you desire.

The last item has to do with setting the height/width of the fragment. There are two ways to this:

- By setting the height and width in the report of the data container. This works best with lists and crosstabs.
- By setting the height and width in the IBM OpenPages GRC on Cloud application when the reporting fragment field is defined. This works best with charts.

Using CrossTracks in Reporting Fragments

Crosstracks are supported in reporting fragments. However, the way we normally create CrossTrack links (with a Hyperlink object in IBM Cognos) does not work from reporting fragments.

In order to obtain crosstrack functionality, you will need to use HTML tags instead of hyperlinks.

Creating a Link to Open in the Same Window

Follow this procedure to create a hyperlink in a reporting fragment that opens in the same browser window.

Tip: In this example we are using the Risk object but you could substitute any object.

Procedure

1. Using the **Query Explorer**, select the main query for the reporting fragment.
2. In your existing list report, drag and drop an HTML Item as a column in your report and do the following:
   a. In the Properties pane, under **HTML Source**, set the **Source Type** property to **Expression**.
   b. In the **Report Expression** property, enter the following:
   ```
   '<a href="' + [DEFAULT_REL].[SOXRISK].[RI_DETAIL_PAGE_URL] + '">' +
       [DEFAULT_REL].[SOXRISK].[RI_NAME00] + '</a>'
   ```
   c. Click **OK**.
3. Select the column title text and do the following:
   a. In the Properties pane, under **Text Source**, set the **Text** property to **Risk Name**.
4. Cut the original Risk Name column from the report.
5. Now you need to make RI_NAME00 a property of the list.
   a. Select the **List** object from the main report page.
   b. In the Properties pane under the **Data** category, double-click on **Properties**.
   c. Select `[RI_NAME00]`.
   d. Click **OK**.
6. Save the report.
7. Have your system administrator configure a reporting fragment field for this report.

Results

When the reporting fragment field is executed in IBM OpenPages GRC on Cloud, it will appear as a cell in a detailed view.

When you click on the CrossTrack, it will bring you to the object detail page in the same window.

Creating a Link to Open in a New Window

Follow this procedure to open the object detail page in a new browser window.

Procedure

1. In the Query Explorer do the following:
   a. Drag and drop a Data Item **Data Item** into the Data Items pane.
   b. In the Properties pane, under the **Data Item** category, double-click on the **Expression** property.
   c. Enter the following for the expression:
      ```
      '<a href="#"' + ' onclick="' + 'javascript:window.open(' + chr(39) +
          [DEFAULT_REL].[SOXRISK].[RI_DETAIL_PAGE_URL] + chr(39) + ')' + '">' +
              [DEFAULT_REL].[SOXRISK].[RI_NAME00] + '</a>'
      ```
   d. Set the **Name** property to **URL Info**.
   e. Set the **Aggregate Function** to **None**.
   f. Set the **Rollup Aggregate Function** to **None**.
   g. Click **OK**.
2. On the main page of your list report, drag and drop an HTML Item HTML Item as a column in your report and do the following:
   a. In the Properties pane, under **HTML Source**, set the **Source Type** property to **Data Item Value**.
   b. Set the **Data Item Value** property to **URL Info**.
   c. In the Properties pane, under **Text Source**, set the **Text** property to **Risk Name**.
3. Cut the original Risk Name column from the report.
4. Now you need to make RI_DETAIL_PAGE_URL a property of the list.
   a. Select the **List** object from the main report page.
   b. In the Properties pane under the **Data** category, double-click on **Properties**.
   c. Select `[RI_DETAIL_PAGE_URL]`**.**
   d. Click **OK**.
5. Save the report.
6. Have your system administrator configure a reporting fragment field for this report.

Results

When the reporting fragment is executed in the IBM OpenPages GRC on Cloud, it will appear as a field in a detailed view.

When you click on the CrossTrack link, it will bring you to the object detail page in a new pop-up window.

Supplying Configuration Fields

In order for the system administrator to create the fragment field in the IBM OpenPages GRC on Cloud application, you must supply the following fields: Report Path, Fragment Name, Object ID Prompt, Reporting Period ID Prompt, Height, and Width.

- **Report Path**: The location of the report within the CommandCenter repository. See “Obtaining the Report Path.”
- **Fragment Name**: The name of the report element to be displayed in the reporting fragment. This is typically the name of a data container. See “Obtaining the Fragment Name.”
- **Object ID Prompt**: The name of the object ID prompt in the report. This is an optional field. See “Obtaining Object Prompts.”
- **Reporting Period ID Prompt**: The name of the reporting period prompt. This is an optional field. See “Obtaining Object Prompts.”
- **Height** and **Width**: The height and width of the display area in pixels. This is typically only used when a list is being displayed. See “Obtaining Height and Width” on page 217.

In order to obtain the above values, the best method is to cut and paste the values from CommandCenter to a text file and send them to your system administrator. Create labels in your text file corresponding to the above fields. Start the text file with a **Name** and **Description** field for your reporting fragment.

Obtaining the Report Path

Follow this procedure to obtain the **Report Path**.

Procedure

1. In the reporting portal, navigate to the folder with the reporting fragment report.
2. Click on the **Set Properties** icon.
3. Click on the **View the search path, ID and URL** hyperlink.
4. In the pop-up, cut and past the **Search Path** into the text file as the **Reporting Path**.
5. Click **Close**.

Obtaining the Fragment Name

Follow this procedure to obtain the **Fragment Name**.

Procedure

1. Select the main page of the report.
2. Select the report element you want to display in the fragment. This will be a list, chart or crosstab.
3. In the **Properties Pane** under **Miscellaneous**, cut and past the **Name** property into your text file.

Obtaining Object Prompts

Follow this procedure to obtain the **Object ID Prompt**.

Procedure

1. If a prompt page does not exist, select the main query subject of the fragment in the **Query Explorer** and find the prompt name in one of the filters. Skip the remainder of this section.
2. Otherwise use the **Page Explorer**, navigate to the prompt page. Select the control that contains the **Object ID Prompt**.
3. In the **Properties Pane** under **General**, cut and paste the **Parameter** property into the text file.

4. Repeat step 3 to obtain the **Reporting Period ID Prompt** (if it exists).

Obtaining Height and Width

The height and width parameters are optional. If the report is a chart, the values can be left blank and the chart will be auto-scaled. If the report is a list or crosstab, determine an optimal **Height** and **Width** to minimize scrolling.

Chapter 18. Custom Query Subjects

Custom query subjects (CQS) are a feature that is offered by IBM OpenPages GRC on Cloud so that a report author can create a user defined SQL statement to solve a problem that cannot be solved through the use of the framework. The out-of-the-box framework model represents an environment that typically solves a majority of a customer's reporting needs; the rest are solved using custom query subjects.

The following describes the process flow for creating reports using a custom query subject

1. Review the report design document.
2. Create an ER diagram of the objects required.
3. Create SQL statements in the SQL development tool.
4. Add OpenPages database functions.
5. Substitute variables.
6. Complete testing in the SQL development tool.
7. Create a new report.
8. Add SQL to a SQL object.
9. Put variables back into the SQL.
10. Complete report development.
11. Create an entry in the CQS file.
12. Regenerate framework.
13. Replace SQL object references with CQS.

Reviewing Your Report Design

The first step in designing a Custom Query Subject is to review your report design. You should pay special attention to the fields and objects required to produce the report. This will enable you to decide whether to create a custom query subject, or use the existing reporting framework model.

Creating an Entity-Relationship Diagram

The next step in creating a custom query subject is to generate an Entity-Relationship (ER) diagram. From IBM OpenPages GRC on Cloud you can determine the fields attached to an object and the relationships between objects.

Go to **Administration | Object types**. This list contains all of the objects defined in your system.

If you select an individual object you can see the field groups defined for the object.

The field group for the object contains the definition of all of the fields defined for the object. This does not include keys and other ID fields that might be needed to create prompts or joins between tables. Refer to Chapter 1, "Database Models," on page 1 for more information on fields required to join tables.

Using this information, you can now determine which objects are required in your SQL statement. As specified in Chapter 1, “Database Models,” on page 1, each one of these objects corresponds to a reporting table and a set of bridge tables. You need to determine the relationship between the objects and thus what type of join is required and which bridge tables to use.

At this point you can create a ER diagram in a diagramming tool which will form the basis of your SQL statement. It is a good practice to review the diagram with one of the system architects to verify its accuracy.

Creating an SQL Statement

Using an SQL development tool, you can now create your SQL statement.

Following the SQL coding standards listed in Appendix A, “SQL Coding Guidelines,” on page 239, write your SQL making sure that all fields that are required for the report are included. Check the result set for duplicate records and proper grouping. When you are done, you will be able to cut and paste this SQL statement into reports.

You need to remember to use the RV_ views when you create your SQL statement. Each RV_ view contains an "OpenPages Security" column, OP_FLAG_READ_ACCESS, with these possible values:

- Y (for yes), if the user running the report has read access to the record.
- N (for no), if the user running the report does not have read access to the record.

For example, if you want IBM OpenPages GRC on Cloud security to be integrated into CQS, you would add the filter on this column:

```
OP_FLAG_READ_ACCESS ='Y'
```

If you are using Oracle SQL Developer, there are instances where SQL Developer does not operate properly because of a problem with not setting the user ID properly to a default value. You can manually set the locale variable by executing the following SQL statement in your session:

```
begin OP_SESSION_MGR.SET_ACTOR_ID_PRIVATE (#$account.parameters.openPagesUserId#);
  end;
```

Where you substitute the IBM Cognos macro for a valid user ID value.

SQL Statements with field level encryption

When you use the PROPERTY.AUDIT.TRAIL table in your SQL query, and field level encryption is enabled, decrypt the property value from the PROPERTY.AUDIT.TRAIL table to allow users to see the clear text values.

The following is the database function for decrypting property values:

- op_crypt_mgr.decrypt_property_value - This is used to decrypt values in the PROPERTY.AUDIT.TRAIL table.

The following are the input properties for the op_crypt_mgr.decrypt_property_value database function:

- Property 1: propertydefs.propertydefid (PropertyDef ID of the field.)
- Property 2: propertyvals.value (Property Value of the field.)

For example, select op_crypt_mgr.decrypt_property_value(t.propertydefid, t.old_value) from property_audit_trail t.

Adding Database Functions

Chapter 13, “Using Predefined Database Functions,” on page 157 describes all of the IBM OpenPages GRC on Cloud database functions that have been exposed to the report author. There are three key functions that you will need to use:

- `OP_ACTOR_MGR.GET_DISPLAY_NAME` - This is used to convert userIds to the actual user names.
- `OP_RPS_AUX.CONVERT_EVID_TO_NAME` - This is used to convert a single select enumerated string to its text.
- `OP_RPS_AUX.CONVERT_EVIDLIST_TO_NAME` - This is used to convert a multi-select enumerated string to a comma separated list of text.

Once you have created your SQL statement you will be able to see in the result sets which items are return as IDs and which are returned as text. You will need to convert the strings returned as IDs to readable text to make your report useful. You should verify that the fields you are converting are actual enumerated strings by going to the **Administrator | Object Types** menu item and searching the fields on the individual objects.

The `GET_DISPLAY_NAME` function is used (based upon the requirements of your particular organization) to show user names in reports as opposed to the login name.

Substituting Variables

After you have included the database functions there will be references to substitution strings that are only available in the IBM Cognos reporting environment. In order to test your SQL you will need to make the following substitutions:

- `#$account.parameters.openPagesLocaleId#` - This is locale of the user who is logged into the system. Use "1" (EN_US).
- `#$account.parameters.openPagesUserId#` - This is the ID number of the user who is logged into the system. Query the `actorinfo` table and use the `actorId` value for the OpenPagesAdministrator instead of this variable.

Testing the SQL Statement

At this point you have created a valid SQL statement that can be tested. You should examine the result set for the following:

- Duplicate records
- Bad joins
- Data that needs conversion
- Missing or wrong fields
- Incorrectly named fields
- Testing performance

Creating the Report

At this point you have a fully functioning SQL statement that contains all of the fields you require to complete your report.

Select a report type to create a new report.

Adding a SQL Object

Follow this procedure to add a SQL object.

1. Open the query explorer and select the query that was created for the report.
2. Select a SQL object from the toolbox menu in the left window pane and drop it onto the query subject next to the existing query subject.
3. Select the SQL object and change the Data Source property to "OpenPages DataSource". This is the name given to the driver that connects to your IBM OpenPages GRC on Cloud database.

Adding SQL to the SQL Object

You can add your code by double clicking on the SQL object and pasting your SQL code into the **SQL** dialog box.

Click the **Validate** button to make sure there are no errors in your SQL code. Click **OK** to continue.

You can now select the query subject from the query explorer and see that the SQL object appears just like a query subject with query items that can be selected.

At this point you can complete your report design using the custom SQL as the basis for your report.

Putting Variables Back Into the SQL

Once the SQL statement is copied into the SQL object, you will need to replace the default values for database functions, security and prompts with the correct variables and prompts.

Adding Prompts

A prompt asks the user for a value during the execution of a SQL statement. A prompt is a macro in IBM Cognos and cannot be executed by SQL development tools. This is why you add prompts after you add the SQL to the SQL object.

The basic syntax for the prompt is as follows:

```
prompt (Name, Data Type, Default Text)
```

The following is the definition of the variables use in the prompt macro:

- **Name** - This is the name of the prompt. This can also be the same name as a parameter on the prompt page. If you use the parameter as the prompt name, the report prompt page will appear. This is required.
- **Data Type** - This is the type of data returned by the prompt. Values include: integer, date and string. This is required.

- **`Default Text`** - This is the default value of the prompt. This is optional.

The following are examples of the syntax for using prompts:

- String Prompt - `#prompt("bundle_name", "string")#`
- Integer Prompt - `#prompt("resourceId", "integer", "1")#`
- Date Prompt - `#prompt("Start Date", "date")#`

Only add prompts to the SQL when you use nested SQL statements. It is always better to put the prompts in the report. Also, when you add the CQS to the xml file, you must add a prompt section to the query.

For more information on using prompts refer to the *IBM Cognos 8 Framework Manger* documentation.

Completing Report Development

At this point you can complete all of your report development using the SQL object as the basis for your report. You may find that as you continue your report development that you are missing fields or need to apply IBM OpenPages GRC on Cloud functions to some of the fields in the query. You can edit the SQL object and add these missing items since you have not yet converted the report to run with a Custom Query Subject.

Creating an Entry in the CQS File

The framework generation contains two parts, the dynamically generated portion and the custom portion.

The dynamic part of the framework is created by the framework generator based on a user's metadata. The custom portion of the framework is contained in two files: `static-modelquerysubjects.xml` and `custom-static-modelquerysubjects.xml`. They contain user-defined query subjects and the relationships between them.

The `static-modelquerysubjects.xml` file is used by IBM OpenPages GRC on Cloud to define custom query subjects including auditing, security, user permission, disassociated file, and checked-out files. This file is shipped with each installation and will be overwritten during upgrade process. We recommend not making any modification to this file.

The `custom-static-modelquerysubjects.xml` is the file used by customers to add their own custom query subjects. This file is not shipped and will not be overwritten during the upgrade process. This file is located on your OpenPages GRC on Cloud server in the `CommandCenter|framework|conf` folder for your IBM OpenPages framework. The default paths are as follows:

- Using the Legacy Model Framework on a Windows operating system: `C:\OpenPages\CommandCenter\framework\conf`
- Using the V6 Framework on a Windows operating system: `C:\OpenPages\CommandCenter\framework\conf\mqs`
- Using the Legacy Model Framework on an AIX or Linux operating system: `/opt/OpenPages/CommandCenter/framework/conf`
- Using the V6 Framework on an AIX or Linux operating system: `/opt/OpenPages/CommandCenter/framework/conf/mqs`

The file containing the CQS is an XML file. The best method to use to add an additional query subject is to copy an existing one and replace all the data. When editing the file you must be careful to create the beginning and end tags properly.

The file is constructed of the following tags:

- `<MODELQUERYSUBJECTS>` - This is the outer most tag of the file. This is a collection of `<MODELQUERYSUBECT>` definitions.
- `<MODELQUERYSUBJECT>` - This is an individual custom query subject.
- `<RELATIONSHIPS>` - This is the relationship between two query subjects.
- `<CALCULATIONS>` - This is a custom calculation.
- `<PARAMETERMAPS>` - This is a user defined parameter map.

The following sections will describe the proper use of the elements of each section of the xml file.

Adding a Model Query Subject

The model query subject is the definition of the SQL statement you want to add to the framework. This contains the definition of where you want to add the query, the query itself, the columns in the query and prompts (if used).

It is a good idea to place a comment before the model query subject indicating the name of the report where the CQS is used and the CQS author.

The following sections list the elements contained in a model query subject, the attributes of those elements, the elements they contain, and the elements in which they are contained.

MODELQUERYSUBJECTS

Contains: MODELQUERYSUBJECT

Contained by: None.

MODELQUERYSUBJECT

Attributes:

`name` Custom query subject name. All query subject names must be unique. You cannot use the "-" in the name of a query subject.

Required.

`namespaces`
The namespace to add the query to. Can be "ALL" or a comma separated list of namespaces. If this field is omitted, "DEFAULT" will be used.

Optional.

`passthrough`
Set to "true" if IBM Cognos optimization will be skipped, "false" if IBM Cognos optimization will be applied.

Optional.

`folder` The folder that will be created/used to store the new custom query subject under each defined namespace. The recommended location is: /MISCELLANEOUS/CUSTOM QUERY SUBJECTS

Optional.

childId

Optional.

parentId

Optional.

Contains: <![CDATA[SQL statement]]>, COLUMNS, PROMPTS, KEYS

Contained by: MODEQUERYSUBJECTS.

<![CDATA[SQL statement]]>

This section of the XML contains the actual SQL expression.

Attributes: None.

Contains: None.

Contained by: MODELQUERYSUBJECT

COLUMNS

Attributes: None.

Contains: column

Contained by: MODELQUERYSUBJECT

column

This element contains all the exposed columns from the actual SQL statement.

Attributes:

name All column names must appear in upper case.

Required.

Contains: None.

Contained by: COLUMNS

PROMPTS

Attributes: None.

Contains: prompt

Contained by: MODELQUERYSUBJECT

prompt

Attributes:

name The prompt name that appears in the query subject. It must match exactly.

defaultvalue
The default value for the prompt.

Optional.

Contains: None.

Contained by: PROMPTS

KEYS

Attributes: None.

Contains: None.

Contained by: MODELQUERYSUBJECT

MODELQUERYSUBJECT Example

```
<!-- Control Owner Assignment Report - Author: J. Doe -->
<MODELQUERYSUBJECT
  name="Control_Owner_Assignment"
  passthrough="false"
  folder="/MISCELLANEOUS/CUSTOM QUERY SUBJECT">
  <![CDATA[
    SELECT
      RA.CHECKED_IN_BY as RA_OWNER,
      RA.NAME00 as RA_NAME,
      RI.CHECKED_IN_BY AS RI_OWNER,
      CN.CONTROL_ID as CN_CONTROL_ID,
      CN.NAME00 as CN_NAME,
      CN.DESCRIPTION as CN_DESCRIPTION,
      CN.CHECKED_IN_BY as CN_OWNER
      FROM
        RV_RISKASSESSMENT RA,
        RT__RISKASSESSMENT_RISK RA2RI,
        RV_RISK RI,
        RT__RISK_CONTROL RI2CN,
        RV_CONTROL CN
      WHERE
```

```
        RA.RISKASSESSMENT_ID = RA2RI.RISKASSESSMENT_ID  AND
        RA.REPORTING_PERIOD_ID = RA2RI.REPORTING_PERIOD_ID AND
        RA2RI.RISK_ID = RI.RISK_ID AND
        RA2RI.REPORTING_PERIOD_ID = RI.REPORTING_PERIOD_ID AND
        RA2RI.IS_PRIMARY = 'Y' AND
        RA2RI.HIER_INSTANCE_ID = 1 AND
        RI.RISK_ID = RI2CN.RISK_ID AND
        RI.REPORTING_PERIOD_ID = RI2CN.REPORTING_PERIOD_ID AND
        RI2CN.CONTROL_ID = CN.CONTROL_ID AND
        RI2CN.REPORTING_PERIOD_ID = CN.REPORTING_PERIOD_ID  AND
        RI2CN.IS_PRIMARY = 'Y' AND
        RI2CN.HIER_INSTANCE_ID = 1 AND
        RA.REPORTING_PERIOD_ID  = #prompt("reportingperiodid", "integer")#
        AND RA.OP_FLAG_READ_ACCESS ='Y'
        AND RI.OP_FLAG_READ_ACCESS ='Y'
        AND CN.OP_FLAG_READ_ACCESS ='Y'
       ]]>
  <COLUMNS>
    <column name="RA_OWNER"/>
    <column name="RA_NAME"/>
    <column name="RI_OWNER"/>
    <column name="CN_NAME"/>
    <column name="CN_DESCRIPTION"/>
    <column name="CN_OWNER"/>
  </COLUMNS>
  <PROMPTS>
    <prompt name="reportingperiodid" defaultValue="-1"/>
  </PROMPTS>
</MODELQUERYSUBJECT>
```

Specifying Relationships

The relationship section is used to create joins between custom query subjects or custom query subjects and pre-defined query subjects. If you are designing a set of queries that are related to each other, that relationship needs to be specified. This portion of the xml file enables you to create those relationships.

The following sections list the relationship elements, the attributes of those elements, the elements they contain, and the elements in which they are contained.

RELATIONSHIPS

Attributes:

Contains: relationship

Contained by: MODELQUERYSUBJECTS

relationship

Attributes:

parent The name of the parent query subject.

Required.

child The name of the child query subject.

Required.

pid The field to join to on the parent.

Required.

cid The field to join to on the child.

Required.

innerjoin
Is this an inner join? Values are "true" or "false".

Optional.

useReportingPeriodId
Whether to join on the reporting period. Values are "true" or "false".

Optional.

one-to-n-left-outer-join
Is this a left outer join? Values are "true" or "false".

Optional.

Contains: None.

Contained by: RELATIONSHIPS

RELATIONSHIPS Example

```
<RELATIONSHIPS>
  <relationship
    parent="RISKEVAL-BY-SOXRISK"
    child="RISK_RATINGS"
    pid="RE_RISKEVAL_ID"
    cid="RE_RISKEVAL_ID"
    useReportingPeriodId="true"
    one-to-n-left-outer-join="true"/>

  <relationship
    parent="RISKEVAL-BY-SOXRISK"
    child="RISK_ALLOCATED"
    pid="RE_RISKEVAL_ID"
    cid="RE_RISKEVAL_ID"
    useReportingPeriodId="true"
    one-to-n-left-outer-join="true"/>

</RELATIONSHIPS>
```

Creating Calculations

You can create custom calculations to provide report authors with calculated values that they regularly use. Calculations can use query items, parameters, variables, calculated members, expressions, and expression components, such as functions.

The following sections list the calculation elements, the attributes of those elements, the elements they contain, and the elements in which they are contained.

CALCULATIONS

Attributes:

Contains: calculation

Contained by: MODELQUERYSUBJECTS

calculation

Attributes:

name
The name of the calculation. Spaces are not allowed.

Optional.

namespace
The namespace in which the calculation object should be created.

Required.

folder
The folder path in which to nest the calculation object. As a standard, the calculation is stored in the calculations folder under the object most closely related to the calculation.

Optional.

level
This is used to help define the order in which calculations are created. A higher level means it will be created later relative to other calculations.

Optional.

Contains: expression

Contained by: CALCULATIONS

expression

Attributes: None.

Contains: <![CDATA[expression]]>

Contained by: calculation

<![CDATA[expression]]>

The SQL expression. It is required.

Attributes: None.

Contains: None.

Contained by: expression

CALCULATIONS Example

```
<CALCULATIONS>
  <calculation
    name = "RISK_LOSS"
    namespace = "DEFAULT_REL"
    folder = "/GRC_OBJECTS/SOXRISK_FOLDER/COMPUTED FIELDS"
    level = "2">
    <expression>
      <![CDATA[nvl(average([DEFAULT_REL].[SOXRISK].
         [RI_INHERENT_SEVERIT_BA] for
        [DEFAULT_REL].[SOXRISK].[RI_RISK_ID]),0)]]>
    </expression>
  </calculation>
</CALCULATIONS>
```

Using Parameter Maps to Create Expressions at Run-Time

Parameter maps are used to create substitutions at run-time. For instance, the ID value of a Risk Category with the name of "Fraud" might be different on your production and test systems. Instead of creating an expression with the numeric ID, you create the expression with the name of the variable and through the use of parameter map, the ID value is filled in at run-time.

The following sections list the parameter map elements, the attributes of those elements, the elements they contain, and the elements in which they are contained.

PARAMETERMAPS

Attributes:

Contains: parametermap

Contained by: MODELQUERYSUBJECTS

parametermap

Attributes:

name The name of the parameter map you wish to create. Spaces are not allowed.

Optional.

Contains: param

Contained by: PARAMETERMAPS

param

Attributes:

key The key that will be substituted for the value.

Optional.

value The value that will be used in the parameter map expression.

Optional.

Contains: None.

Contained by: parametermap

PARAMETERMAPS Example

```
<PARAMETERMAPS>
  <parametermap name = "CUSTOM_RISK_CATEGORY">
    <param key = "Internal Fraud" value = "2345"/>
    <param key = "External Fraud" value = "2346"/>
    <param key = "Employee Theft" value = "2347"/>
  </parametermap>
</PARAMETERMAPS>
```

Regenerating the Framework

Follow the procedures in the *IBM OpenPages GRC on Cloud Administrator's Guide* to update the framework. You will only need to check the Custom Query Subjects and Labels options.

Often during the generation of the framework, you will receive error messages. The following are the most common types of errors:

- You have forgotten to add a column tag for each field in the SQL statement.
- You have a typo in one of the attributes of the MODELQUERYSUBJECT.
- You are missing a closing tag.
- You forgot to define a prompt tag for each prompt in the SQL code.
- Number of column statements should be the same as the fields in the SQL statement.
- The column names in the SQL statement are not spelled the same as in the column tags.
- The column names should be in the same order as the column tags.
- You cannot use "-" in the name of the CQS.
- The limit of characters for field names is 30 characters (this will truncate the field name).

The detail of the error messages can be found in the `script-player-error.log` file located on your report server in `C:\OpenPages\CommandCenter\framework\logs`

If there is a parsing error detected in the file, the error will be listed in the `framework-generator.log` in the same directory.

Note: AIX, Linux, and Windows have different default locations. Also the Legacy framework and 6.0 framework have different file locations.

Replacing SQL Object References with CQS References

At this point you will need to remove the references to the SQL object in the report and replace them with the references to the new custom query subject. You can do this one field at a time by opening every variable in the report and changing the `[SQL<N>]` object references to the name of the new CQS `[<CQS Name>]`.

You can also save the report to a text file and do a global search and replace on `[SQL<N>]` with `[<CQS Name>]` in the file.

You then need to delete all the SQL objects that you created because they are not being used.

Edit the Report Text File

Follow this procedure to save the report to a text file and do a global search and replace on `[SQL<N>]` with `[<CQS Name>]` in the file. Once this is done, reload the file and validate it to make sure your edits were correct.

Procedure

1. Copy the report to the clipboard using the **Tools | Copy Report to Clipboard** menu item.
2. Open a text editor paste the clipboard into the document.
3. Search on `[SQL<N>]` and replace it with `[<CQS Name>]`.
4. Save the file.
5. Select the entire file.
6. Copy the report to the clipboard.
7. Save the existing report.
8. Open a new blank report.
9. Using the **Tools | Open Report from Clipboard**, restore the report from the clipboard.
10. Test that the report functions.
11. Save the report.

Remove SQL Object

Follow this procedure to delete all the SQL objects that you created because they are not being used.

Procedure

1. In your report, open the **Query Explorer** and select **Queries**.
2. Select the **SQL Object** you want to remove.
3. Click the × **Delete** icon on the tool bar.
4. Click the ☑ **Validate** icon. If you did the substitution correctly, the report should still validate.

Chapter 19. Migrating Reports to the V6 Framework

This chapter contains a basic procedure for migrating reports from the legacy framework to the V6 framework and covers all of the known issues.

Report Migration Process

The migration of reports from the legacy product framework to the IBM OpenPages GRC on Cloud 7.0.0 framework requires knowledge and understanding of both frameworks. In addition, you are migrating ReportNet 1.1 reports to an IBM Cognos 10 reporting environment and must understand those differences also.

For instance, in the legacy framework, the product created relationships between primary and secondary objects along with "by" relationships for the report authors. In the V6 framework, the model has been simplified leaving only the relationships between the primary objects in a namespace. It is then left to the report author to create the secondary relationships with the join object in IBM Cognos 10.2. This is just one of the many differences between the two frameworks that must be understood.

The following is a typical process.

1. “Importing the Report into IBM Cognos 10.2”
2. “Changing Framework Packages” on page 234
   - “Updating the Default Namespaces” on page 234
   - “Updating Data Item Names” on page 234
3. “Updating Query Relationships/Names” on page 235
4. “Updating the Conditional Formulas” on page 235“Creating Joins” on page 236
5. “Updating Multi-Byte Characters” on page 237
6. “Setting Unions to Auto Generate” on page 237
7. “Updating the Report Style” on page 237
8. “Updating Multi-Select Text Boxes” on page 238
9. “Removing Extra Query Subjects” on page 238

Importing the Report into IBM Cognos 10.2

The first step in the process is to copy the report from your existing environment to the new IBM Cognos 10 environment. Do the following:

Procedure

1. Open the report in your existing ReportNet 1.1 environment.
2. Validate the report. If errors occur, fix them before you attempt to migrate the report.
3. Click on **Tools | Copy Report To Clipboard**.

   Now you will open the report in your IBM OpenPages GRC on Cloud environment.
4. Login into IBM Cognos 10 and create a blank report.
5. Click on **Tools | OpenReport From Clipboard.**
6. Your report has now been imported to IBM Cognos 10 and the report definition converted to the current xml format.

7. Save your report.

Changing Framework Packages

In IBM OpenPages GRC on Cloud there exist two packages: IBM OpenPages GRC on Cloud Reports (for the legacy framework) and IBM OpenPages GRC on Cloud Reporting V6 for the new framework.

You must now move your report from one framework to the other. Do the following:

Procedure

1. Click on **File | Report Package**.
2. In the dialog select the radio button **Another Package** and then the **OPENPAGES_REPORTS_V6** package.
3. Click **OK**.
4. The report will fail the validation.
5. Click **Close** to remove the validation dialog box.
6. Save your report.

Updating the Default Namespaces

In addition to the package being different the structure of the namespaces is different in the V6 package. In the legacy package there was only one default namespace OPENPAGES_DEFAULT. In the V6 package all namespaces have two default sub-namespaces DEFAULT_REL and DEFAULT_DIM representing the relational model and the dimensional model.

You must now migrate the namespaces in your report to the new namespaces. The easiest way to do this is to edit the report file itself. Do the following:

Procedure

1. Determine which namespaces are utilized in your report by looking at report data items. The namespace will be the first element in the data identifier. In the following example, we can see the namespace is OPENPAGES_DEFAULT.
   ```
   [OPENPAGES_DEFAULT].[SOXCONTROL].[CN_NAME00]
   ```
2. The OPENPAGES_DEFAULT namespace translates to the DEFAULT_REL namespace in the new model.
3. Copy the report to the clipboard using the **Tools | Copy Report to Clipboard** menu item.
4. Open a text editor such as NotePad or WordPad.
5. Paste the clipboard buffer (which contains the report) into the editor.
6. Move the cursor to the first character in the file.
7. Perform a search and replace on [OPENPAGES_DEFAULT] to [DEFAULT_REL].
8. Select the entire file and copy it to the clipboard.
9. Replace the existing report by using the **Tools | Copy Report From Clipboard** menu item.
10. Save the report as a new temporary version. (i.e., report -V1).

Updating Data Item Names

Once you have completed this step, you must go manually edit all data items where the name has been changed in the new framework. Do the following:

Procedure

1. Run a validation on the new report.
2. Click on **Query Details**.
3. What this dialog shows is each data item that does not exist in the new framework. You will need to find each one of these data items and edit the definition.
4. All [SOXBUSENTITY] data items need to change to [SOXBUSENTITY_GPC].
5. All [XXX_RPS_ID] field names (where XXX is GEN, PEN or CEN) need to change to [GEN_REPORTING_PERIOD_ID].
6. Any enumerated string whose names have changed will need to be edited.

Updating Query Relationships/Names

Another major difference you will find between ReportNet 1.1 and IBM Cognos 10 has to do with how it represents the structure of queries in the query explorer.

In ReportNet 1.1 you have a pictorial representation of the structures.

When you update a report from ReportNet to IBM Cognos 10, the pictorial structure no longer exists.

In addition, the conversion eliminated a redundant tabular model TMParent. The report will still function but the pictorial representation needs to be fixed so future updates to this report can be performed more easily. Do the following:

Procedure

1. Open the **Query Explorer** and click on **Queries**.
2. Select **TMChild** and drag it next to **Heat_Map**.
3. Now select **TMGC** and drag it next to **TMChild**.
4. Right mouse click on the pane and select **Expand References.**
5. The queries are now corrected linked.
6. At this point the query names should be updated to the current naming standard.

Updating the Conditional Formulas

IBM Cognos 10 is more particular about how conditional formulas are created. In ReportNet 1.1 you could create a data item in a query subject that could be used in a conditional formula regardless of what data types were used. This can still be done in IBM Cognos 10 with one caveat: there can be no data conversion.

For instance, if your conditional formula was originally as follows:

```
Case
  when ([Percent Effective] > 0.8]) then ('Green')
  when ([Percent Effective] > 0.6]) then ('Yellow')
  else ('Red')
end
```

This formula converts [Percent Effective] (a number) to a color (a string) which is no longer allowed. You must either copy the formula from the data item to the conditional formula or change the formula as such:

```
Case
  when ([Percent Effective] > 0.8]) then (1)
  when ([Percent Effective] > 0.6]) then (2)
  else (3)
end
```

The above formula no longer converts a number to a string.

In addition we have seen issues where spaces have been added to text strings in conditional statements. If you had the following formula in ReportNet 1.1:

```
Case
  when ([Residual Risk Rating] = 'High']) then ('Green')
  when ([Residual Risk Rating] = 'Medium']) then ('Yellow')
  else ('Red')
end
```

We have seen this converted to the following:

```
Case
  when ([Residual Risk Rating] = 'High']) then ('Green')
  when ([Residual Risk Rating] = 'Medium']) then ('Yellow')
  else ('Red')
end
```

You will notice that extra spaces have been added to the Green and Red values to make them the same length as Yellow. This conversion will not work and the extra spaces need to be removed.

Creating Joins

In the IBM OpenPages GRC on Cloud model there are tables that no longer exist. These tables were eliminated to both simplify and speed up the execution of the model. However when you convert reports that utilized these missing tables, you will need to perform a detailed analysis of why you were joining to this table and determine how you are going to replace it.

A good example can be seen in the Entity to Issue relationship. In the old framework there existed a query subject [OPENPAGES_DEFAULT].[ALL-SOXISSUE-UNDER-SOXBUSENTITY] which was the indirect relationship between all entities and all issues. That relationship no longer exists, you must create it within IBM Cognos 10.

In order to do this, do the following:

Procedure

1. Create a query called **Entity** with all the fields from the **Entity** object that are required.
2. Create a query called **Issue** with all the fields from the **Issue** object that are required including the parent context fields for entities.
3. Drop a third query called **listMain** into the report that will be the join of the previous two queries.
4. Drop a join object into the report next to the third query and then drop the other two queries into the join.
5. Double click on the join object.

6. For a direct join (give me only issues under any entity object) do the following:
   a. Click **New Link**.
   b. Select **REPORTING_PERIOD_ID** for Issue and **CEN_REPORTING_PERIOD_ID** for Entity.
   c. Click New Link.
   d. Select **CEN_ENTITY_ID** for Issue and **CEN_ENTITY_ID** for Entity.
   e. Change the cardinality on the Issue to **0..n** (left outer join).
   f. Click **Ok**.
7. For an indirect join (give me all Issues under any object under any entity) do the following:
   a. Click **Convert to expression**.
   b. Enter the following formula:
      ```
      "OP_RPS_AUX.IS_REL_PARENT_CHILD"(
          [Entity].[CEN_ENTITY_ID],
          [Issue].[IS_ISSUE_ID],
          [Entity].[CEN_REPORTING_PERIOD_ID]) = 'Y'
      ```
   c. Click **Ok**.

Results

Note: Once a join is converted to an expression, the only way to get back to the visual representation is by deleting the expression and clicking Ok. The next time you open the join the Join Relationships dialog will show up.

Updating Multi-Byte Characters

If you have localized your report and used any Asian or Unicode characters sets, these character strings might not update properly when the conversion is performed. The fields will be filled in with question marks to indicate they were not migrated properly. You will have to edit these fields one at a time to fix them.

Setting Unions to Auto Generate

A new feature has been added to the definition of unions that enables you to manually define the output variables of a union instead of using the variables from the first query in the union.

We have found that during the conversion process the **Project List** property gets set to **Manual** and the variables selected are not the same as the ones previously selected in ReportNet 1.1. This will break your report. You need to reset this property to **Automatically generated** to enable your unions to function in the same manner they did in ReportNet 1.1.

Updating the Report Style

Once you have the report validated, you might want to update the following style items to the current standards for your organization:

- Change the header and footer - This is an appropriate time to change the style of the headers and footers on all of your reports to reflect any changes in the company branding.
- Update the default header color - The default background cell header color in IBM Cognos 10 is B7D2E2. The upgrade process will not change this color from the default in the older versions of Report Studio.
- Add the username to the report - Previous versions of your reports might not have included the name of the user who ran the reports.

- Replace any lists in headers with singletons - In the older versions of IBM Cognos you were required to place a list data container in the header to display any item that was not a prompt. In IBM Cognos 10 you can use a singleton which is more efficient.
- Change icons/images to relative path - If you had previously been using an absolute path to display icons and image (`/crn/skins/branding/icon-SOXRisk.gif`) you need to switch to a relative path (`../skins/branding/icon-SOXRisk.gif`) to accommodate future upgrades.

Updating Multi-Select Text Boxes

Under certain conditions, multi-select text boxes will not upgrade properly from ReportNet 1.1.

Procedure

1. The prompt is defined as a text box.
2. The text box is set to multi-select.
3. In the Properties pane, under **Positioning**, the **Size and Overflow** property is set.

Results

You are unable to use this prompt because the input box is too narrow. The solution to this problem is to select the **Size and Overflow** property in the **Properties** pane and remove the width. Deleting the value in the width will correct the problem.

Removing Extra Query Subjects

During the upgrade process we have seen CommandCenter add extraneous query subjects to the reports. These query subjects don’t affect the functionality of the report but they do affect the performance of the report.

Procedure

1. In **Query Explorer**, select queries.
2. Delete the TMParent2 query subject.
3. Temporarily rename TMChild to TMParent 2 and connect it to TMParent. You are doing this in order to attempt enable the fields in TMParent (which currently reference TMParent 2) to pick up the references from TMChild.
4. Rename TMParent 2 back to TMChild.
5. Open the TMParent query subject and verify that each data item now references TMChild instead of TMParent 2.
6. If this method failed, the references will still be to the TMParent 2 query subject. You would then need to edit each data item in TMParent by hand to point to TMChild instead of TMParent 2.

Appendix A. SQL Coding Guidelines

This topic describes some general information to follow when working with SQL statements in the custom query subject file.

General Guidelines

Use the following guidelines with SQL statements in the custom query subject file.

- All indentations must be saved as spaces not tabs.
- The indentation must consist of 3 spaces.
- Every nested statement must begin with new level of indentation.
- All database reserved words will be in uppercase.
- The length of lines should not exceed 80 characters.
- Comments should be used in the SQL to describe functionality. The comments must be in the form of `/* <text> */`.
- Filters and grouping should not be placed in the CQS unless absolutely necessary. These should be performed in the report.
- Use UNION ALL instead of UNION, it is much faster. UNION will remove any duplicate rows, UNION ALL will not. Duplicate rows can be removed in the report through the use of the Auto Group and Summarize property.
- You will use the RV_ views instead of the RT_ tables because the views have user security built into them.

Naming Conventions

The mnemonics listed in this topic will be used as both table names and as prefixes for column names.

The Alternate Table Prefix is used when the Field/Table Prefix is a database reserved word.

Table 72. Naming Conventions

Object Name	Object Label	Object Prefix	Alt. Prefix
Assertion	Assertion	AO	
Attestation	Attestation	AN	
AuditableEntity	Auditable Entity	AE	
Auditor	Auditor	AD	
AuditPhase	Audit Section	AH	
AuditProgram	Audit	AU	
Campaign	Campaign	CP	
CostCenter	Cost Center	CC	
CtlEval	Control Eval	CV	
Employee	Employee	EE	
Finding	Finding	FD	
Incident	Incident	IN	INC
KeyPerfIndicator	KPI	KP	

Table 72. Naming Conventions (continued)

Object Name	Object Label	Object Prefix	Alt. Prefix
KeyPerfIndicatorValue	KPI Value	KY	
KeyRiskIndicator	KRI	KR	
KeyRiskIndicatorValue	KRI Value	KE	
LossEvent	Loss Event	LE	
LossImpact	Loss Impact	LO	
LossRecovery	Loss Recovery	LR	
Mandate	Mandate	MD	
ORXLoss	ORX Loss	OL	
Plan	Plan	PN	
Policy	Policy	PL	
PolicyReviewComment	Policy Review Comment	RP	
Preference	Preference	PF	
PrefGrp	Preference Group	PG	
Procedure	Procedure	PC	
ProjectActionItem	Milestone Action Item	PA	
Qsection	Section	QE	
Quest	Question	QF	
Questionnaire	Questionnaire	QD	
RAEval	Risk Assessment Eval	AV	
RegApp	Regulation Applicability	RB	
RegChange	Regulatory Change	RD	
RegInt	Regulator Interaction	RF	
RegTask	Regulatory Task	RT	
Regulator	Regulator	RE	
Requirement	Requirement	RQ	
Resource	Resource	RU	
ResourceLink	Resource Link®	RL	
ReviewComment	Audit Review Comment	RC	
RICat	RI Category	RH	
RIReq	RI Request	RR	
RiskAssessment	Risk Assessment	RA	
RiskEntity	Control Plan	RN	
RiskEval	Risk Eval	RV	
RiskSubEntity	Baseline	RS	
ScenarioAnalysis	Scenario Analysis	BS	
SOXAccount	Account	AC	
SOXBusEntity	Business Entity	EN	
SOXControl	Control	CN	
SOXControlObjective	Control Objective	CO	
SOXDocument	File	FI	

Table 72. Naming Conventions (continued)

Object Name	Object Label	Object Prefix	Alt. Prefix
SOXExternalDocument	Link	LI	
SOXIssue	Issue	IS	ISU
SOXMilestone	Milestone	MI	
SOXProcess	Process	PR	
SOXProject	---	PO	
SOXRisk	Risk	RI	
SOXSignature	Signature	SI	
SOXSubaccount	Sub-Account	SU	
SOXSubprocess	Sub-Process	SB	
SOXTask	Action Item	AT	ATI
SOXTest	Test Plan	TE	
SOXTestResult	Test Result	TR	
Submandate	Sub-Mandate	SM	
Timesheet	Timesheet	TI	
Waiver	Waiver	WV	
Workpaper	Workpaper	WP	

SELECT Clause

Use the following guidelines when using SELECT clauses.

- Place each field in the SELECT clause on its own line.
- Do not use wildcard statements in the SELECT clause. For example,

  ```
  SELECT * from RV_ENTITY
  ```

  will select all fields from the entity view but will be slow.
- SELECT field values will always use aliases:

  ```
      PR.NAME00 as PR_NAME
  ```
- All report variables will be prefixed with the object name. For example, Process Name will appear in the report as PR_NAME. Table 72 on page 239 contains the acceptable prefixes and alternates when the prefix is a database reserved word.
- If the CQS is to be joined with other Query Subjects, one of the columns must be called REPORTING_PERIOD_ID. It cannot have a prefix.
- Always include key (RESOURCE_ID, REPORTING_PERIOD_ID) columns from each of the object type (RV_) tables as a field in the SELECT clause. There is no performance price to pay and they are useful for filtering the report.
- Always include the IS_PRIMARY and DISTANCE Columns from each of bridge (RT_) tables in the SELECT clause. There is no performance price to pay and it is useful for filtering the report.
- Use SQL sub-selects to convert single value enumerated strings and username conversions. For example, use the following to obtain an issue status value:

  ```
          (SELECT
              EV.LCLTXT_NAME
           FROM
              I18N_INT_ENUM_VALS EV
           WHERE
              EV.ENUM_VAL_ID  = ISS.STATUS AND
  ```

```
            EV.I18N_LOCALE_ID = #$account.parameters.openPagesLocaleId#
         ) AS IS_STATUS
Use the following to obtain the first and last names from a userId:
         (SELECT
            UI.FIRSTNAME||' '||UI.LASTNAME
         FROM
            ACTORINFO ACI,
            USERINFO UI
         WHERE
            PR.ITRM_RISK_MGR = ACI.NAME AND
            ACI.ACTORID = UI.ACTORID
         ) AS PR_RISK_MANAGER_NAME
```

- Use the IBM OpenPages GRC on Cloud CONVERT_EVIDLIST_TO_NAME database function to converted multivalued enumerated strings to lists.

FROM Clause

Use the following guidelines when using FROM clauses.

- Place each table in a FROM clause on its own line.
- List tables in the FROM clause in the order in which they will be accessed.
- Table names will be also be aliased with mnemonics from Table 72 on page 239. For example the RV_PROCESS view will be:

  RV_PROCESS as PR

  the Incident view will be

  RV_INCIDENT as INC
- Bridge tables will be named with the from and to objects, for example Business Entity to Process, EN2PR.

WHERE Clause

Use the following guidelines when using WHERE clauses.

- Use the ANSI-92 format of joins:

  ```
  EV.REPORTING_PERIOD_ID (+)= EN.REPORTING_PERIOD_ID
  ```

  for a left outer join. This is to improve the readability of the SQL.
- You should use indexed columns in joins or when filtering. This will create higher performance SQL.
- Every join should be an INNER join if possible. Never use CROSS JOIN, only INNER and OUTER ones.
- Also avoid searching with not equals operators (<> and NOT) as they result in table and index scans.
- Avoid wildcard characters at the beginning of a search on an indexed column using the LIKE keyword. This results in an index scan which is slow. The first example results in an index scan, while the second example results in an index seek:

  ```
  SELECT entity_id FROM rv_entity as EN WHERE en.full_path LIKE '
        %Financial Services'
  '%Financial Services' AND EN.RA.OP_FLAG_READ_ACCESS ='Y'
  SELECT entity_id FROM rv_entity as EN WHERE en.full_path LIKE
        '/%Financial Services'
  '/%Financial Services' AND EN.RA.OP_FLAG_READ_ACCESS ='Y'
  ```
- Typical joins will be by the resource ID (object Id) and reporting period ID. These are both indexed fields.

- Only use the HEIR_INSTANCE_ID = 1 in a join when you are both joining to a bridge table and from a bridge table. These joins will make use of the <parent>_ID, <child>_ID, and REPORTING_PERIOD_ID and will force the use of the system index on this table.
- If you are using the IS_PRIMARY field, do not use the HEIR_INSTANCE_ID field.

ORDER BY Clause

Use the following guideline when using ORDER BY clauses.

- Do not use the column numbers in the ORDER BY clause, only use column names.

Appendix B. Namespace Definitions

This appendix contains the definitions of all of the IBM OpenPages GRC on Cloud out-of-the-box namespaces as of the release date of this document. Depending on your system configuration and what modules are being used, not all of these namespaces will be available.

These namespaces have been used to support the out-of-the-box reports that have been supplied with your system. The namespaces can be reused for additional reporting requirements and edited to meet your needs. Please be aware that if you edit a namespace by deleting or changing the relationship between objects, the out-of-the-box reports may no longer function.

The most accurate definition of your namespaces can be found in the registry settings for your system. Under the menu item **Administration | Settings**, go to the folder **OpenPages | Platform | Reporting Framework V6 | Models | OPENPAGES_FRAMEWORK_V6 | Namespaces** to view your system's namespace definitions.

The following is the list of the out-of-the-box namespaces:

- AUD1 - Audit 1 namespace.
- AUD2 - Audit 2 namespace.
- DEFAULT - The Default namespace used by all Modules.
- ITG1 - IT Governance 1 namespace.
- ITG2 - IT Governance 2 namespace.
- MAND1 - Mandate 1 namespace.
- MAND2 - Mandate 2 namespace.
- ORM1 - Operational Risk Management 1 namespace.
- ORM2 - Operational Risk Management 2 namespace.
- ORM3 - Operational Risk Management 3 namespace.
- POL1 - Policy 1 namespace.
- RA1 - Risk Assessment 1 namespace.
- RA2 - Risk Assessment 2 namespace.
- REGAPP1 - Regulation Applicability 1 namespace.

AUD1

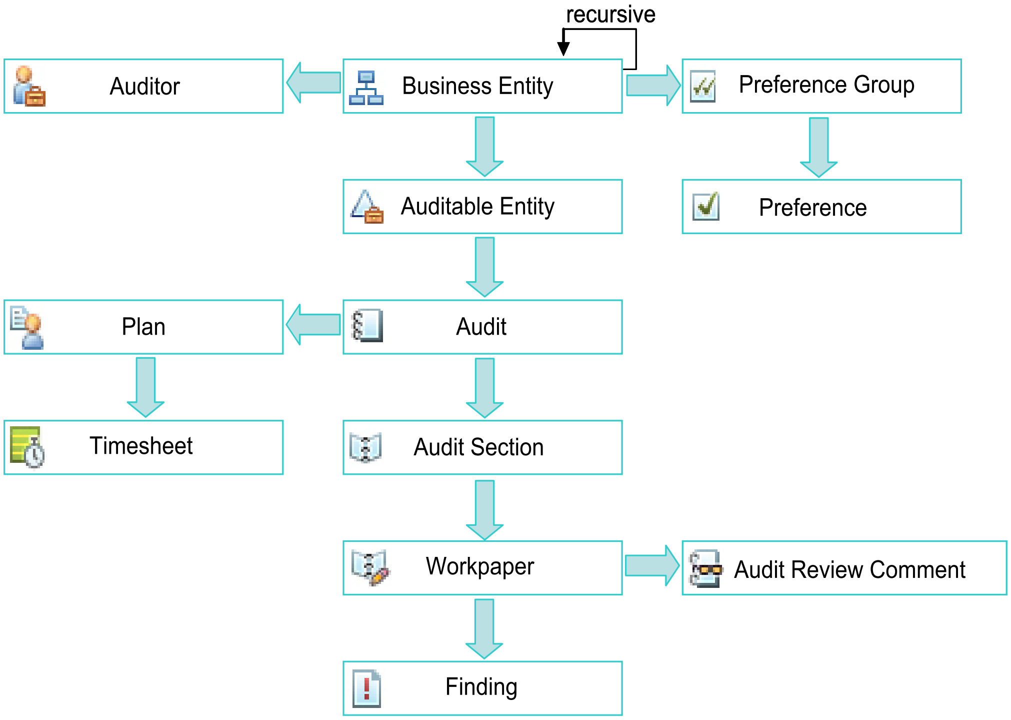


Figure 113. Audit 1 Namespace

AUD2

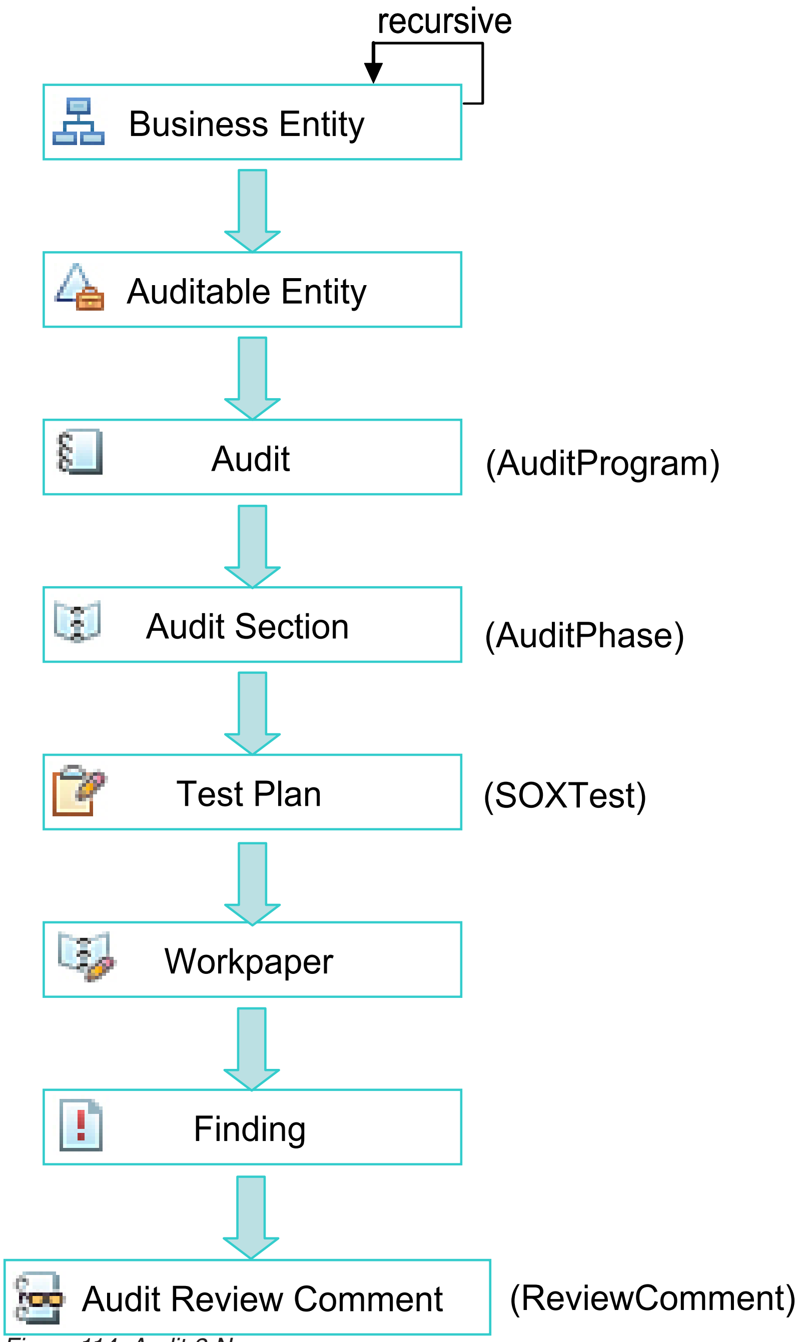


Figure 114. Audit 2 Namespace

DEFAULT

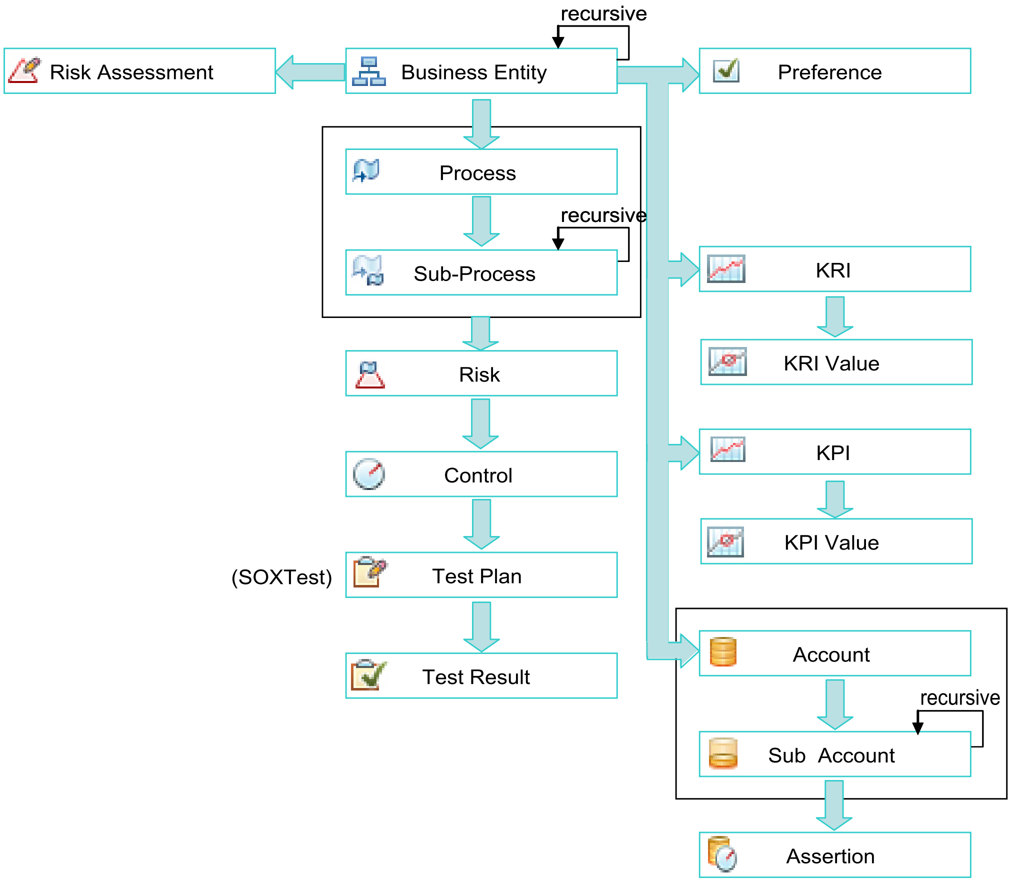


Figure 115. Default Namespace

ITG1

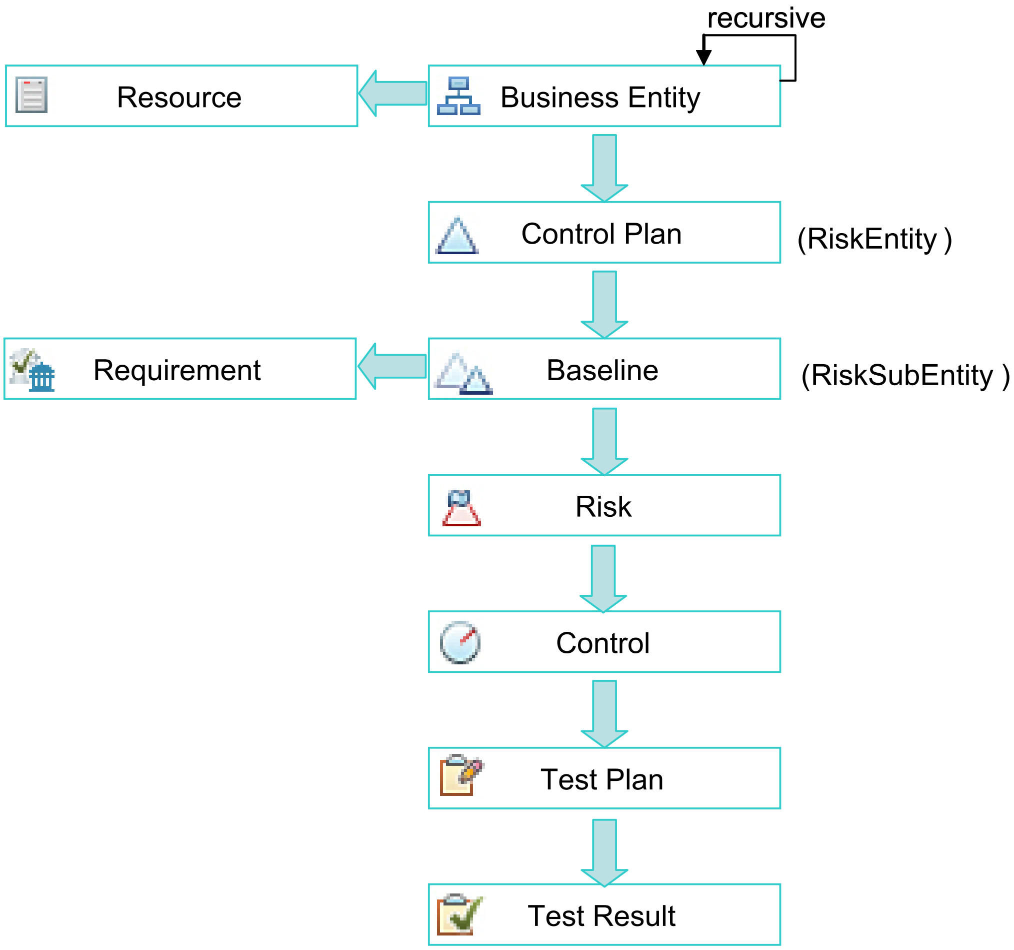


Figure 116. IT Governance 1 Namespace

ITG2

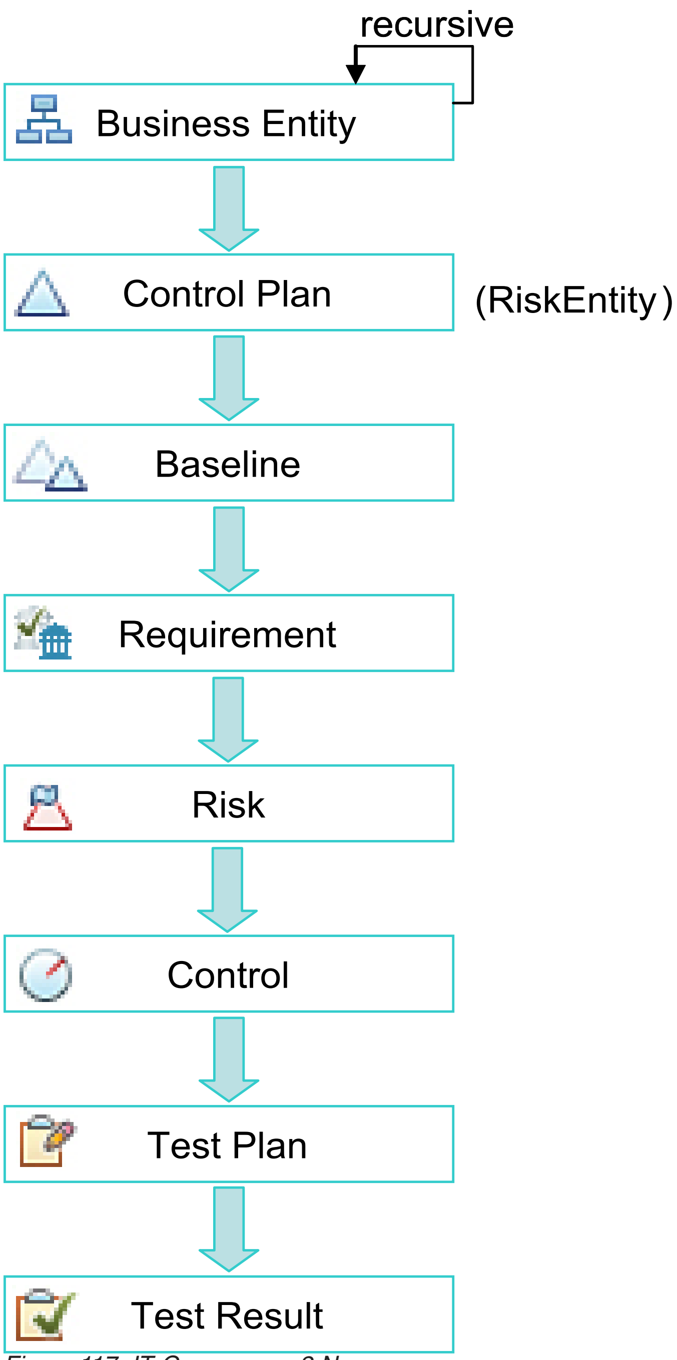


Figure 117. IT Governance 2 Namespace

MAND1

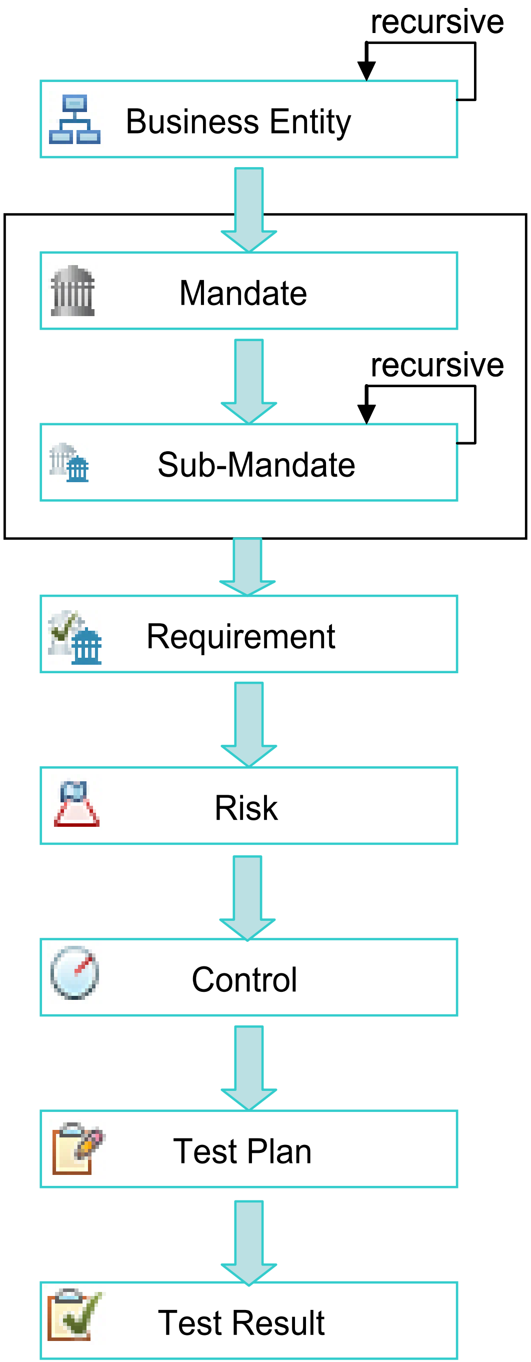


Figure 118. Mandate 1 Namespace

MAND2

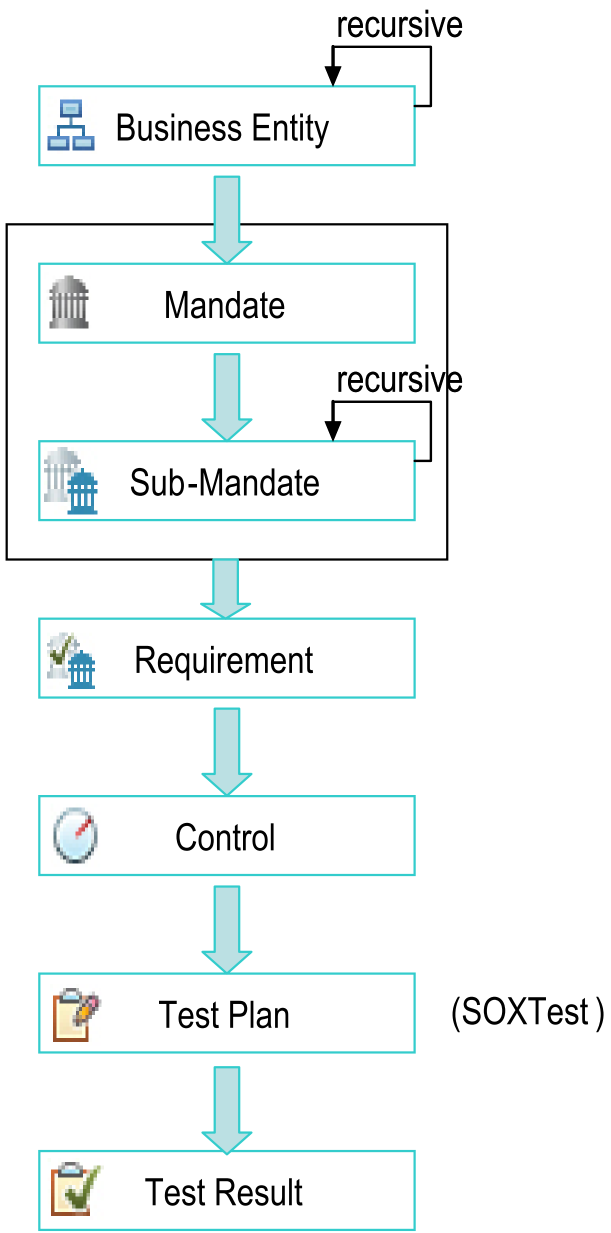


Figure 119. Mandate 2 Namespace

ORM1

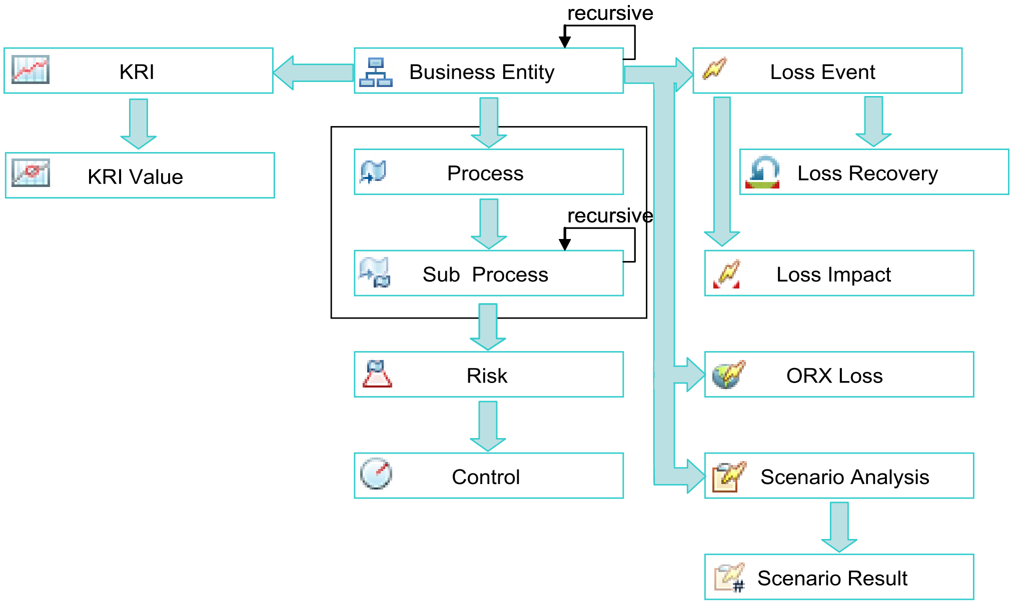


Figure 120. Operational Risk Management 1 Namespace

ORM2

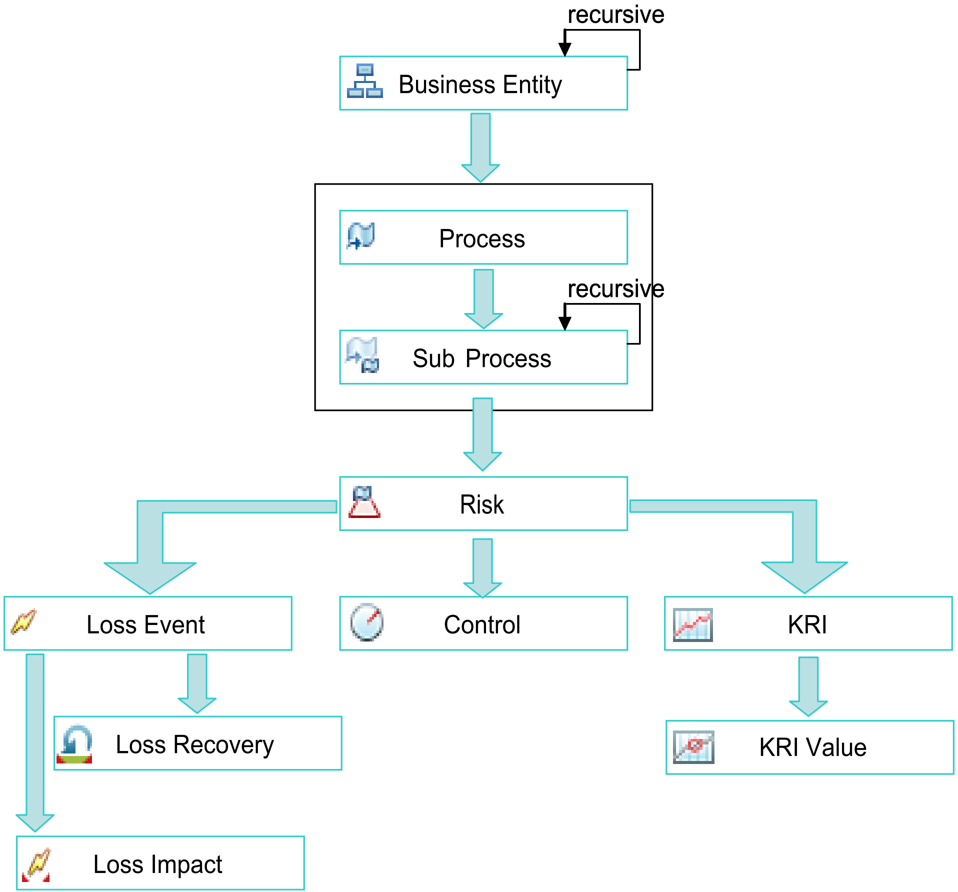


Figure 121. Operational Risk Management 2 Namespace

ORM3

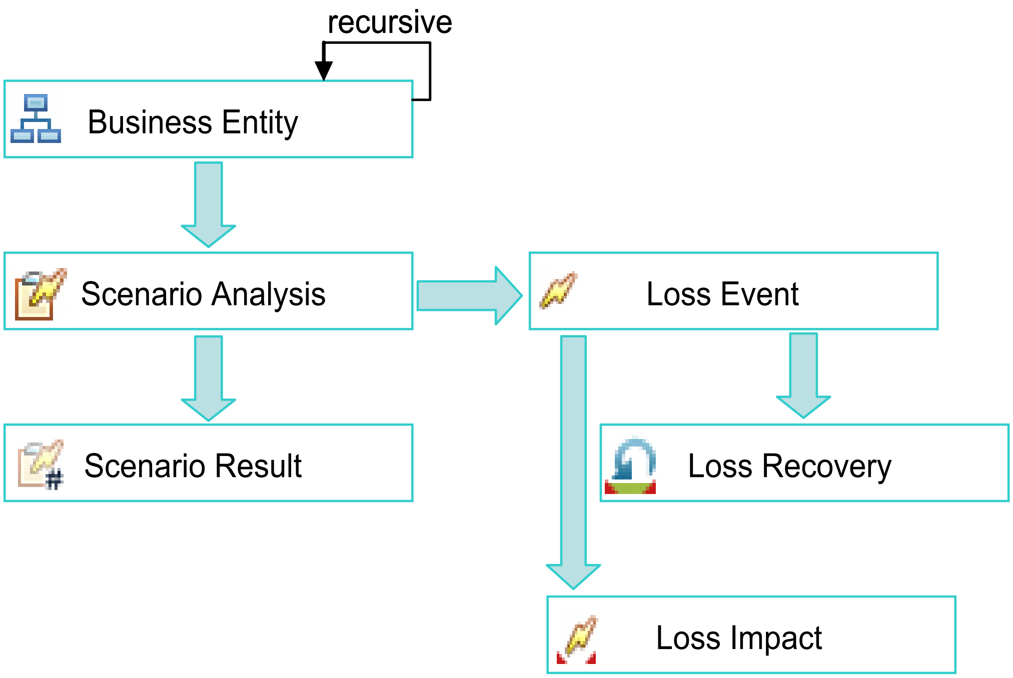


Figure 122. Operational Risk Management 3 Namespace

POL1

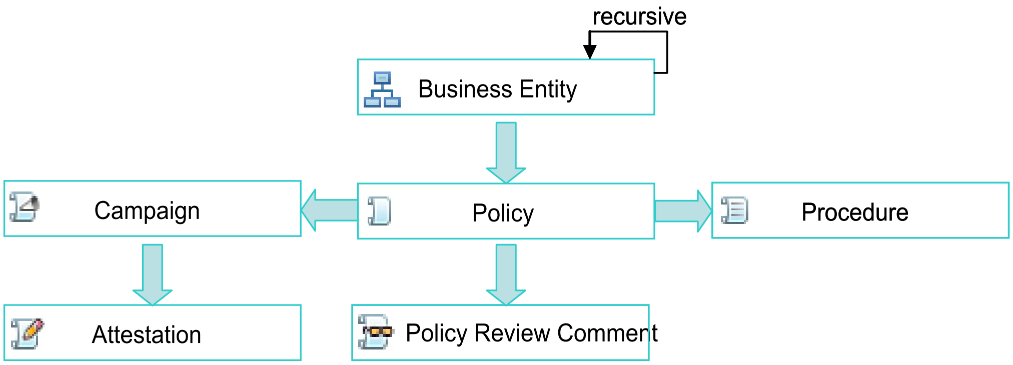


Figure 123. Policy 1 Namespace

RA1

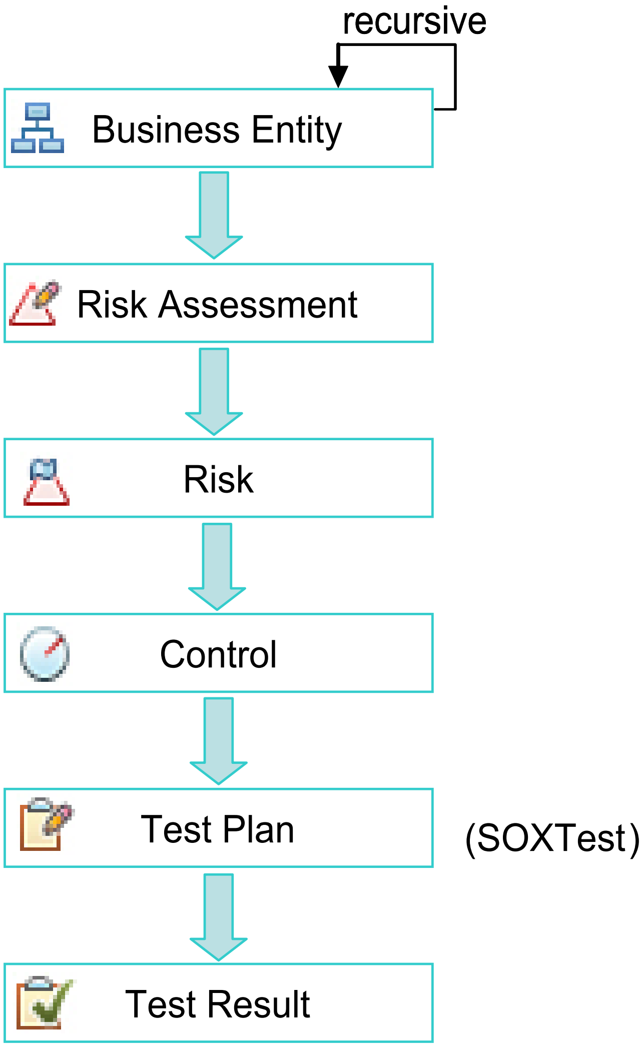


Figure 124. Risk Assessment 1 Namespace

Note: The Risk Eval object is disabled out-of-the-box and might need to be enabled depending on your system requirements.

RA2

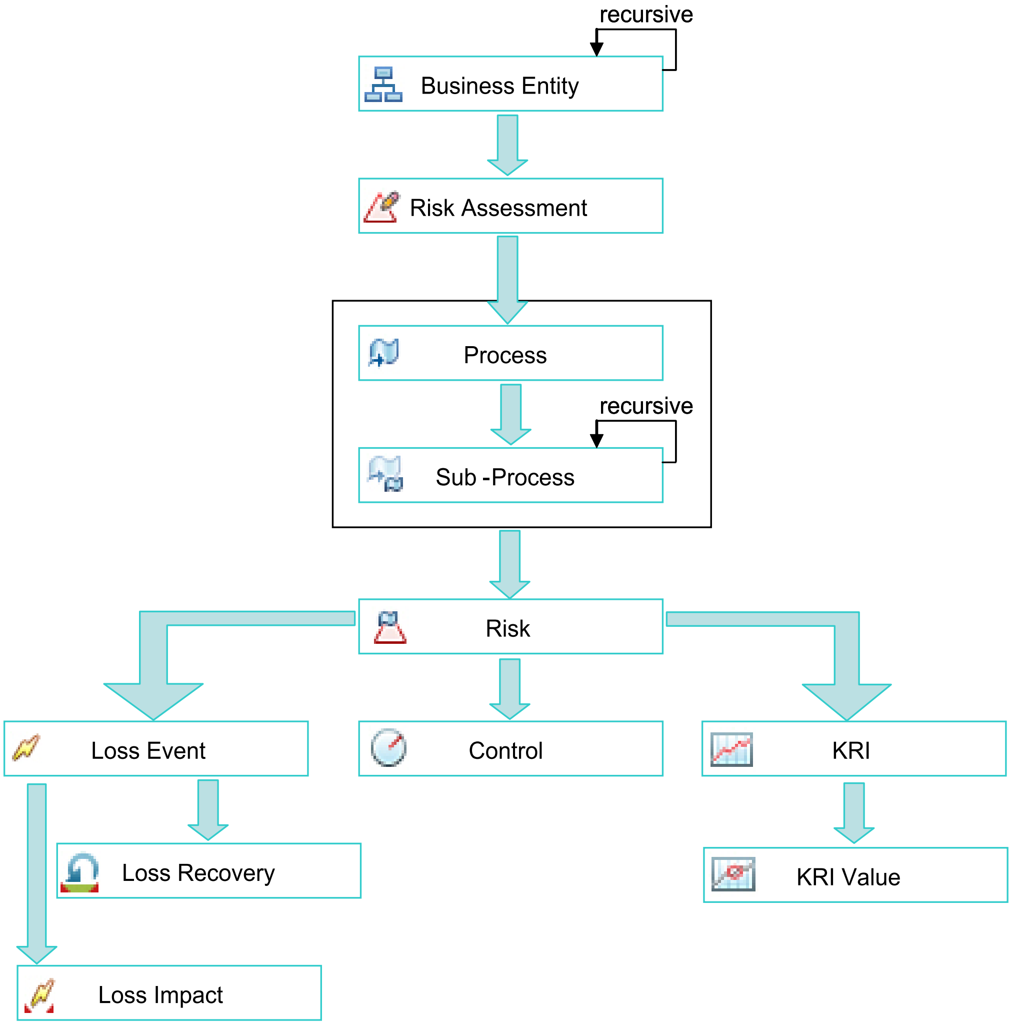


Figure 125. Risk Assessment 2 Namespace

Note: The Risk Eval object is disabled out-of-the-box and might need to be enabled depending on your system requirements.

REGAPP1

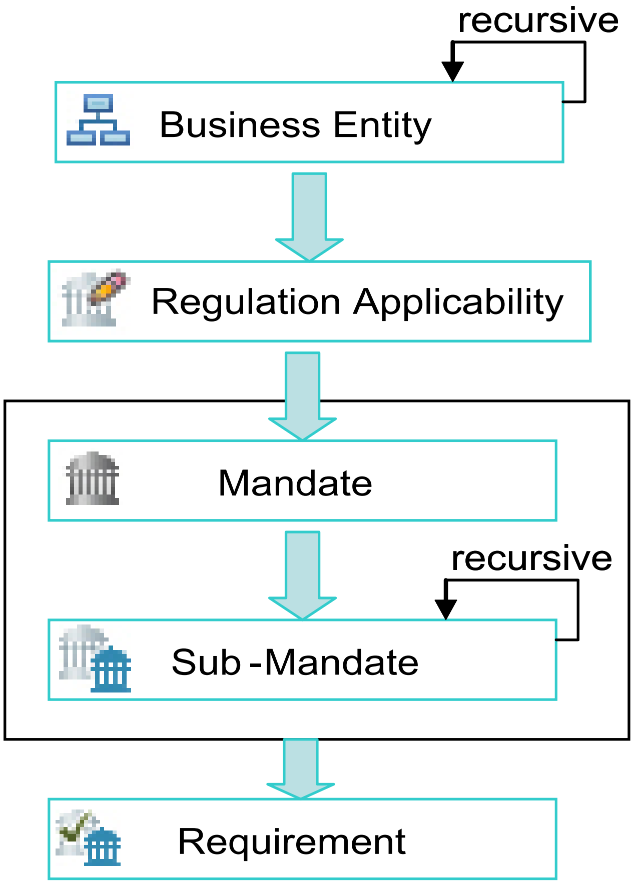


Figure 126. Regulation Applicability 1 Namespace

Appendix C. Limitations and Workarounds

This appendix lists known issues and limitations that report authors may encounter when creating reports, and any possible workarounds.

Equation Length Limitation

There is a limitation on the size of the computation attribute value that can be stored by the IBM OpenPages GRC on Cloud application. The main attribute of concern is 'Equation' where a complex equation could be very lengthy. There is a 20,000 byte limit on the size of the entered text. Note that OpenPages GRC on Cloud supports multi-byte characters and so this may not be the equivalent of 20,000 characters if you are using a multi-byte language.

Workaround

You can create your calculation in the CQS file.

Localization Display Issues

If you are designing reports for use with multiple languages, occasionally format issues such as the following may arise:

- A horizontal scroll bar may be displayed in the report window. For example, all supplied IBM OpenPages GRC on Cloud reports are designed to display generated data in a set window size. When some of these reports are localized to other languages, such as German or French, the page width increases and a horizontal scroll bar appears.

Workaround

The following are some workaround suggestions that may fix the horizontal scroll issue:

- Set the feature within Report Studio to allow automatic word breaks.
- Manually break a long word into smaller portions by using, for example, hyphen and space characters (the word will break after the space). You may need to experiment with manually breaking the word so that it displays properly within the column.

Related concepts:

"Language and locale support" on page 43
If you are using IBM OpenPages GRC on Cloud in a language other than English, this information will help you to understand the language and locale settings.

Error when grouping large string fields

If you group a large string field with an unspecified length, either implicitly or explicitly, the grouping results in an unindexed column error. Depending on your report, you might see an error similar to this text:

```
Using a blob column in this query requires that the query subject
[TABLES].[PROPERTYVALS_CLOB] must have either a key or a unique index.
```

An example of an explicit grouping would be if you set the **Auto Group & Summarize** query property to *Yes* for a query that includes a large string field. An example of an implicit grouping would be if you add a grouping on a large string field in a list report.

Workaround

If you want to group a large string field, you must first cast it into a character data type with a maximum length supported by your database. An example of casting would be:

```
cast ([MAND1_REL].[REQUIREMENT].[RQ_SUPPORTING_REQUIREMENT], varchar
(4000))
```

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 Corporation
Location Code FT0
550 King Street
Littleton, MA
01460-1250
U.S.A.

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.

This Software Offering does not use cookies or other technologies to collect personally identifiable information.

Copyright

Licensed Materials - Property of IBM Corporation.

US Government Users Restricted Rights – Use, duplication or disclosure restricted by GSA ADP Schedule Contract with IBM Corp.

This information contains sample application programs in source language, which illustrate programming techniques on various operating platforms. You may copy, modify, and distribute these sample programs in any form without payment to IBM, for the purposes of developing, using, marketing or distributing application programs conforming to the application programming interface for the operating platform for which the sample programs are written.

These examples have not been thoroughly tested under all conditions. IBM, therefore, cannot guarantee or imply reliability, serviceability, or function of these programs. You may copy, modify, and distribute these sample programs in any form without payment to IBM for the purposes of developing, using, marketing, or distributing application programs conforming to IBM's application programming interfaces.

Trademarks

IBM, the IBM logo and ibm.com are trademarks or registered trademarks of International Business Machines Corp., registered in many jurisdictions worldwide.

The following terms are trademarks or registered trademarks of other companies:

- Microsoft, Windows, Windows NT, and the Windows logo are trademarks of Microsoft Corporation in the United States, other countries, or both.
- Linux is a registered trademark of Linus Torvalds in the United States, other countries, or both.

Other product and service names might be trademarks of IBM or other companies. A current list of IBM trademarks is available on the Web at “ Copyright and trademark information ” at www.ibm.com/legal/copytrade.shtml.

Index