CA ERwin® Data Modeler

Creating Custom Mart Reports

Release 9.5.0

Contact CA Technologies

Understanding your Support

Review support maintenance programs and offerings.

Registering for Support

Access the CA Support online registration site to register for product support.

Accessing Technical Support

For your convenience, CA Technologies provides easy access to "One Stop" support for all editions of CA ERwin Data Modeler, and includes the following:

- Online and telephone contact information for technical assistance and customer services
- Information about user communities and forums
- Product and documentation downloads
- CA Support policies and guidelines
- Other helpful resources appropriate for your product

For information about other Home Office, Small Business, and Enterprise CA Technologies products, visit http://ca.com/support.

Provide Feedback

If you have comments or questions about CA Technologies product documentation, you can send a message to techpubs@ca.com.

If you would like to provide feedback about CA Technologies product documentation, complete our short customer survey, which is also available on the CA Support website, found at http://ca.com/docs.

CA ERwin Data Modeler News and Events

Visit www.erwin.com to get up-to-date news, announcements, and events. View video demos and read up on customer success stories and articles by industry experts.

Contents

Chapter 1: Introduction

CA ERwin DM Version 9 uses an XML-based interface that allows XML-based reporting tools to retrieve information from the CA ERwin DM Version 9 Mart. Use the XML-based interface to report on the following objects that are stored in the Mart.

- Catalog Information
- Model objects and their properties
- Users
- Lock information
- Profiles
- Permissions

This section contains the following topics:

Architecture

When users generate a report, CA ERwin Version 9 Mart retrieves data from the Mart Server via web services.

The components involved in reporting are as follows:

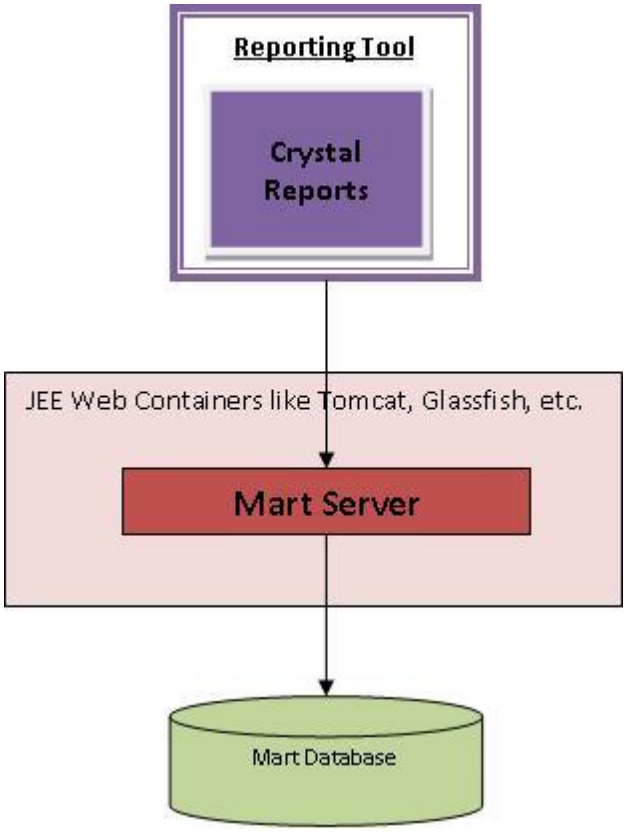
- A Reporting tool that can report on XML data; for example Crystal Reports.
- Mart Server, which is a component of CA ERwin Version 9 Mart installation.
- Mart database, which is the data source for the Mart Server.

CA ERwin DM Release 8 Mart reported directly against the Mart database. This approach is changed in CA ERwin DM Version 9 Mart.

The following steps describe the process of generating reports in CA ERwin DM Version 9 Mart:

1. Reporting tools such as Crystal Reports connect to the Mart Server through web services.
2. The web services return the report information in XML data format.
3. Reporting tools use this XML information as the dataset and display the report.

The following diagram illustrates the reporting architecture:


Reporting Tool
Crystal Reports
JEE Web Containers like Tomcat, Glassfish, etc.
Mart Server
Mart Database

Generate Reports using Crystal Reports

This section describes how to generate reports using Crystal Reports.

Before attempting to generate a report, help ensure that at a minimum you have the View permissions on the Mart on which you want to report.

Follow these steps:

1. Start the web server where you have installed Mart Server, for example, start Tomcat.
2. Use the Crystal Reports Report Wizard and create a connection that uses XML and Web Services as the data source.
3. Use the HTTP(S) Data Source.
4. Follow the Report Wizard steps.
5. Use the following format to enter the URL for HTTP(S) XML URL:

   ```
   http://<Server Name>:<Port
   No>/MartServer/service/report/generateReport/<Report
   Name>/<Username>/<Password>
   ```

   For example:

   ```
   http://localhost:8084/MartServer/service/report/generateReport/Users/sa/erwin
   ```

6. Use the following format to enter the URL for HTTP(S) Schema:

   For example:

   ```
   http://<Server Name>:<Port
   No>/MartServer/service/report/generateSchema/<Schema
   Name>/<Username>/<Password>
   ```

   For example:

   ```
   http://localhost:8084/MartServer/service/report/generateSchema/Users/sa/erwin
   ```

   The URL details are as follows:

   - <Server Name>: Machine name where Mart Server is running.
   - <Port No>: Port number on which J2EE container is running.
   - <Report Name>: Name of the report which being created.
   - <Schema Name>: Name of the report schema which is being created.
   - <Username>: Username for a server user in Mart Server.
   - <Password>: Password for a server user in Mart Server.

   Enter the report URL and schema URL in the browser Address bar to see the correct data. If the username and/or password is incorrect or if you do not have the required authorization to View Reports on the Mart Server, the message, Logon Failed appears.

7. After the new connection is successfully created, expand it.
8. Select the tables for which you want to generate a report and create the report.

Chapter 2: Creating a Custom Mart Report

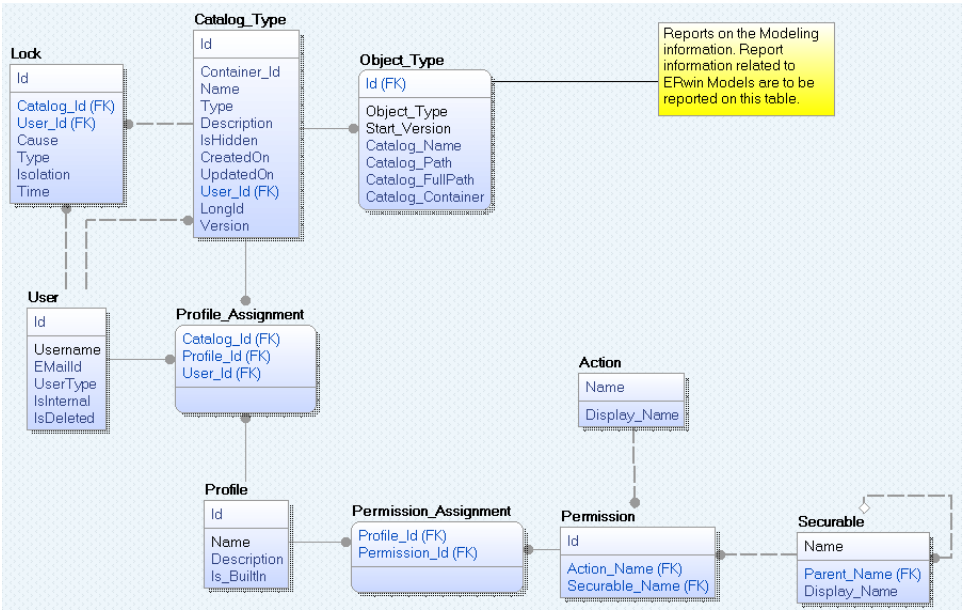
This section contains the following topics:

Create Custom Mart Reports

CA ERwin DM Version 9 Mart lets you create custom Mart reports. A custom Mart report requires two XML files--the report definition file and the report schema file. The report definition file includes names of the tables and columns that you want to include in the report. The report schema file includes how you want the tables and columns to appear in the report. Sample definition and schema files are included as part of the installation files.

This section describes how to edit the sample files to add new or modify existing reports.

The basic tables and columns to report are organized as shown below:



Follow these steps to generate a new report:

1. Define the report definition.
2. Define the report schema.
3. Execute it using a reporting tool.

Define a New Report

The reports.xml file includes report definitions for all the Mart reports. This file is available in the MartServer\WEB-INF folder. Edit this file to modify an existing report definition or add a new report definition.

Report definitions have the following characteristics:

- Report definitions are created in XML format.
- Reports have a unique name that identifies the report.
- Reports are defined using XML elements.

The basic XML elements and tags are shown in the example below.

Follow these steps:

1. Open the MartServer\WEB-INF\Reports.xml file.
2. Enclose report definition within the <report> element.
3. Define a name for the report within the <Name> element.
4. Define each table that you want to include in the report. In the example below, the report is generated on the User table.
5. Enclose the properties or columns of the required table that you want to appear in the output within the <Report_Output> element.
6. Include all properties to report on within the <Property> element.
7. Define each column to report on within the <Type> tag.
8. Close all the tags.
9. (Optional) execute this report in a web browser. Use the URL for reports described in the previous procedure.

An example of defining a report on Users is shown below:

```
<report>

        <Name> Users </Name>

        <User>

                <Report_Output>

                        <Property>

                                <Type>ID</Type>

                                <Type>Username</Type>

                                <Type>EmailID</Type>
```

```
                                        <Type>UserType</Type>
                                        <Type>IsInternal</Type>
                                        <Type>IsDeleted</Type>
                                    </Property>
                                <Report_Output>
                            </User>
                        </report>
```

Define the Report Schema

A schema defines the expected output. The report schema for all Mart reports is defined in the reports-schema.xml file. This file is available in the MartServer\WEB-INF folder. Help ensure that the report data output always confers to the schema definition.

Report schemas have the following characteristics:

- Report schemas are created in XML format.
- Report schemas have a unique name that identifies the schema. We recommend that you have this name same as that of the report name.
- Reports are defined using XML elements.

Follow these steps:

1. Open the MartServer\WEB-INF\Reports-schema.xml file.
2. Enclose report schema definition within the <report_schema> element.
3. Define a name for the report schema within the <Name> element.
4. Define the Schema within the <schema> element in the CDATA section.
5. Defining a schema definition within the CDATA section helps XML accept the XML related information within an XML definition.

Note: There are various tools to generate Schema Definition (i.e. XSD) information from a XML. You can generate the schema definition using a tool of your choice and include it in the reports-schema.xml file.

An example of defining a report schema on Users is shown below.

```
<report_schema>

<Name> Users </Name>

        <schema>

        <Report_Output>

                <![CDATA[<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">

                        <xs:element name="report_root">

                                <xs:complexType>

                                <xs:sequence>

                                        <xs:element name = "User"
maxOccurs="unbounded">

                                        <xs:complexType>
```

```
                                        <xs:sequence>
                                        <xs:element name = "Id" type="xs:string"/>
                                        <xs:element name ="Username"
type="xs:string"/>
                                        <xs:element name ="EmailId"
type="xs:string"/>
                                        <xs:element name ="UserType"
type="xs:string"/>
                                        <xs:element name = "IsInternal"
type="xs:string"/>
                                        <xs:element name = "IsDeleted"
type="xs:string"/>
                                        </xs:sequence>
                                        </xs:complexType>
                                        </xs:element>
                                    </xs:sequence>
                                    </xs:complexType>
                                </xs:element>
                            </xs:schema>]]>
                    </schema>
            </report_schema>
```

Chapter 3: Types of Reports

This section contains the following topics:

Report on Catalogs

The following types of Catalog objects are present in Mart:

- Mart
- Library
- Model
- Version

Follow these steps:

1. Edit the reports.xml file and include the report definition for the Catalog objects and their properties that you want to report on.
2. Edit the reports_schema.xml file and include the schema definition for the report. You can also generate the schema definition using a tool of your choice and include it in the reports_schema.xml file.

   The following report definition script reports on the Catalog table that includes Model and Versions:

```
<report>

    <Name>Library Model Version</Name>

        <Catalog_Type>

            <Name>Model</Name>

            <Report_Output>

                <Property>

                    <Type>Catalog_Name</Type>

                    <Type>Catalog_Path</Type>

                </Property>

            </Report_Output>

            <Catalog_Type>

                <Name>Version</Name>

                <Report_Output>

                    <Property>

                        <Type>Catalog_Name</Type>

                    </Property>
```

```
            </Report_Output>
        </Catalog_Type>
    </Catalog_Type>
</report>
```

The list of Type tags or valid column values are listed below:

Column Name	Description
Id	Unique identifier for the catalog
Name	Name of the catalog item
Type	Type of the catalog
Container_Id	Owner of the catalog item
Catalog_Name	Name of the catalog item
Catalog_Path	Complete path for the catalog item not including the name of the catalog item in context
Catalog_FullPath	Complete path for the catalog item including the name of the catalog item in context
Catalog_Container	Name of the owner of the catalog item
Description	Description of the catalog
CreatedOn	Date and time identifying the catalog creation
UpdatedOn	Date and time identifying the catalog latest update
User_Id	User identifier responsible for creating the catalog item
LongId	Identifier for the catalog
Version	Version number for the catalog item

Report on Model Objects

You can report on modeling objects using the object names and property names. The list of object names is available in the file MartServer\WEB-INF\Metadata\EMX_ObjectTypecodeList.csv. The list of property names is available in the file MartServer\WEB-INF\Metadata\EMX_PropertyTypecodeList.csv.

Follow these steps:

1. Edit the reports.xml file and include the report definition for the Model objects and their properties that you want to report on.
2. Edit the reports_schema.xml file and include the schema definition for the report. You can also generate the schema definition using a tool of your choice and include it in the reports_schema.xml file.

The following report definition script reports on the Object table that includes Entity and Attributes:

```
<report>

        <Name>Entity Attributes</Name>

        <Object_Type>

        <Name>Entity</Name>

            <Report_Output>

                <Property>

                    <Type>Name</Type>

                     <Type>Physical_Name</Type>

                     <Type>Catalog_Name</Type>

                     <Type>Catalog_Path</Type>

                </Property>

            </Report_Output>

            <Object_Type>

                <Name>Attribute</Name>

                <Report_Output>

                    <Property>

                        <Type>Name</Type>
```

```
                    <Type>Physical_Name</Type>
                    <Type>Logical_Data_Type</Type>
                    <Type>Physical_Data_Type</Type>
                </Property>
            </Report_Output>
        </Object_Type>
    </Object_Type>
</report>
```

The list of Type tags or valid column values are listed below.

Column Name	Description
Catalog_Container	Identifier for the catalog item mapped into the object table
Id	GDM identifier for the object
Object_Type	Type of the object
Property	Here we mention the property name to report on for example to report on GDMTypes::pName mention Name here
Catalog_Name	Name of the catalog item
Catalog_Path	Complete path for the catalog item not including the name of the catalog item in context
Catalog_FullPath	Complete path for the catalog item including the name of the catalog item in context
Catalog_Container	Name of the owner of the catalog item
Start_Version	The version that the object was created on

Additional Support for Object Type Reporting

Apart from reporting on Objects and Properties, you may need to generate other reports that need different XML elements.This section describes how you can generate reports for special requirements.

Alias

When generating report data in XML, the element name is the same as that defined in the <Name> element. There may be cases when you require a different name for the element name. In such cases, use the <Alias>Alias_Name< /Alias> tag to set the name of the element as that defined in the Alias in the output XML. Use this name when you refer to the element in the schema and not the name defined within the <Name> element.

Syntax:

```
<Alias>Alias_Name</Alias>
```

All Objects Support

To report on an object type, use that object type within the <Name> element. To report on all objects in Mart, do not define the <Name> element.

In the example script given below, the type of object for which the report is generated is not mentioned. This script generates a report on all objects returning the object type, name, and definition. Remember to add the Alias tag.

Syntax:

```
<report>
    <Name>Definitions</Name>
        <Object_Type>
            <Alias>Objects</Alias>
            <Report_Output>
                <Property>
                    <Type>Object_Type</Type>
                    <Type>Name</Type>
                    <Type>Definition</Type>
                </Property>
             </Report_Output>
         </Object_Type>
</report>
```

Sub Object Type Support

For reporting on a specific object type in a specific context, we include the <Object_Type> tag within the parent <Object_Type> element. The supports within the sub object type support are as follows:

Child

When you define an object type within an existing object type, the child items of that specific type are reported. For example, if you report on an object type that reports an entity that defines an object type of Attribute, then the report includes all attributes within that entity. This is the default way in which Mart reporting is implemented.

Syntax:

```
<report>

    <Name>Entity Attributes</Name>

        <Object_Type>

            <Name>Entity</Name>

            <Report_Output>

                <Property>

                        <Type>Name</Type>

                </Property>

            </Report_Output>

            <Object_Type>

                <Name>Attribute</Name>

                <Report_Output>

                        <Property>

                                <Type>Name</Type>

                </Property>

                </Report_Output>

            </Object_Type>

        </Object_Type>

</report>
```

Owner

To refer to the Owner object in the context of an object type, use the Relationship attribute that refers to "Owner". For example, when you report on an object type that includes a domain and defines an object type of Model with relationship attribute as "Owner," then the report is generated on the Model owning the Domain.

Syntax:

```
<report>
 <Name>Domains</Name>
  <Object_Type>
   <Name>Domain</Name>
       <Report_Output>
               <Property>
                       <Type>Name</Type>
               </Property>
       <Report_Output>
  <Object_Type Relationship = "Owner">
   <Name>Model</Name>
       <Report_Output>
               <Property>
                       <Type>Name</Type>
               </Property>
       <Report_Output>
   </Object_Type>
   </Object_Type>
</report>
```

Referenced Object

To refer to an object that is referenced using a reference property within the object in context, use the Relationship attribute as "Ref" and define another attribute Reference that indicates the reference property to consider on the object while retrieving the object type defined within the current object type element. For example, within an object type reporting on Attribute defining an object type of Domain with relationship attribute as "Ref" and the Reference attribute defined as "Parent_Domain_Ref" would report on the Attribute and the Domain referenced by the property "Parent_Domain_Ref" on the Attribute in context.

Syntax:

```
<report>

  <Name>Attribute Domains</Name>

   <Object_Type>

    <Name>Attribute</Name>

        <Report_Output>

                <Property>

                        <Type>Name</Type>

                </Property>

        <Report_Output>

   <Object_Type Relationship = "Ref" Reference = "Parent_Domain_Ref">

    <Name>Domain</Name>

        <Report_Output>

                <Property>

                        <Type>Name</Type>

                </Property>

        <Report_Output>

    </Object_Type>

    </Object_Type>

</report>
```

Applying Conditions

Sometimes, you apply conditions to filter data. When you apply conditions, the efficiency of reports is improved, as conditions remove the invalid objects in context in the output XML. The conditions are applied on the object in context. As given in the example below, the <Condition Compare="Comparison_Strategy"> tag is defined within the <Object_Type> tag. The "Compare" attribute defines the comparison strategy.

The comparison strategy defines the following strategies:

- Exact: Compares with the exact value defined in the <Value> tag.
- BeginsWith: Value begins with the value defined in the <Value> tag.
- EndsWith: Value ends with the value defined in the <Value> tag.
- Contains: Value contains with the value defined in the <Value> tag.
- NotEqual: Value does not match the value defined in the <Value> tag.
- Null: Property value is NULL. The <Value> tag is not required in this strategy.
- NotNull: Property value exists. The <Value> tag is not required in this strategy.

Syntax:

```
<Object_Type>
 <Name>Attribute</Name>
     <Condition Compare = "Exact">
             <Property>
                     <Type>Tpye</Type>
                     <Value>0</Value><!--0 = PK Attribute Type-->
             </Property>
     </Condition>
     <Report_Output>
             <Property>
                     <Type>ID</Type>
                     <Type>Parent_Relationship_Ref</Type>
             </Property>
     </Report_Output>
</Object_Type>
```

Property Type Alias

When generating report data in XML, the element name for a property is the same as that defined in the <Type> element. There may be cases when you require a different name for the element name; in such cases, use the <Type Alias ="Alias_Name" >Logical_Data_Type<Type> tag to set the name of the element as that defined in the Alias in the output XML. Use this name to refer to the element in the schema and not the name defined within the <Type> element.

Syntax:

```
<Type Alias = "Datatype">Logical_Data_Type</Type>
```

Property Inheritance Value

There may be a scenario when the property on an object is inherited. You can retrieve the inherited property value using the code example given below. The Reference attribute within the <Type> element defines the reference property to look into. The Object_Type attribute defines the corresponding object, whereas the Property attribute defines the property in the object to probe. The inheritance is drilled down until the property value is not null. For example, to retrieve the Name for Domain or Attributes where the Name property on the current domain or attribute is NULL then it gets the "Parent_Domain_Ref" on the current object, on that id it looks into for an object of type "Domain" with the id associated with "Parent_Domain_Ref" property and gets its property "Name". If it is NULL, it further looks into the "Parent_Domain_Ref" property on that object and likewise looks further up till the property is retrieved.

Syntax:

```
<Type Reference = "Parent_Domain_Ref" Object_Type = "Domain" Property =
"Name">Name</Type>
```

UDP Reports

To generate a report on User Defined Properties (UDP) in Mart, use UDP in the <Name> element within the <Object_Type> attribute.

Follow these steps:

1. Edit the reports.xml file and include the report definition for the properties of UDPs that you want to report on.
2. Edit the reports_schema.xml file and include the schema definition for the report. You can also generate the schema definition using a tool of your choice and include it in the reports_schema.xml file.

The following example shows the report definition to report on UDPs and their properties:

Syntax:

```
<report>
 <Name>User-Defined Properties</Name>
  <Object_Type>
   <Name>Udp</Name>
       <Report_Output>
              <Property>
                     <Type>Name</Type>
                     <Type>tag_Udp_Default_Value</Type>
                     <Type>tag_Udp_Owner_Type</Type>
                     <Type>Catalog_Path</Type>
                     <Type>Catalog_Name</Type>
              </Property>
       <Report_Output>
      </Object_Type>
</report>
```

Report on Users

This section describes how you can create the report definition for users.

Follow these steps:

1. Edit the reports.xml file and include the report definition for the properties of the User table that you want to report on.
2. Edit the reports_schema.xml file and include the schema definition for the report. You can also generate the schema definition using a tool of your choice and include it in the reports_schema.xml file.

The following example shows the report definition to report on the User table:

```
<report>
    <Name>Users</Name>
        <User>
            <Report_Output>
                <Property>
                        <Type>ID</Type>
                        <Type>Username</Type>
                        <Type>EMailId</Type>
                        <Type>UserType</Type>
                        <Type>IsInternal</Type>
                        <Type>IsDeleted</Type>
                </Property>
            </Report_Output>
        </User>
</report>
```

Column Name	Description
Id	Unique identifier for the user
Username	Username
EMailId	Email address for the corresponding user

UserType	Type of the User. The types basically are Server User Windows User Group User
IsInternal	Integer "1" indicating if the user is an internal user i.e. a windows user which was authenticated on being part of a group user
IsDeleted	Integer "1" indicating if the user is deleted and no long a valid Mart user

Report on Locks

This section describes how you can create the report definition for locks.

Follow these steps:

1. Edit the reports.xml file and include the report definition for the properties of the Lock table that you want to report on.
2. Edit the reports_schema.xml file and include the schema definition for the report. You can also generate the schema definition using a tool of your choice and include it in the reports_schema.xml file.

The following example shows the report definition to report on the Lock table:

```
<report>
    <Name>Locks</Name>
        <Lock>
            <Report_Output>
                <Property>
                        <Type>ID</Type>
                        <Type>Catalog_ID</Type>
                        <Type>Type</Type>
                        <Type>Time</Type>
                        <Type>User_ID</Type>
                </Property>
            </Report_Output>
        </Lock>
</report>
```

Column Name	Description
Id	Unique identifier for the lock
Catalog_Id	Catalog identifier for which the lock is applicable
User_Id	User identifier that has acquired the lock

Cause	The identifier of the lock that has caused this lock
Type	Type of the current lock. The values: E: Existence S: Shared U: Update X: Exclusive
Isolation	Isolation level of the current lock. The values: C: This lock not only affects the current catalog but also all child entries of the catalog L: This lock only affects the current catalog
Time	Date and time when the lock was applied

Report on Profiles

This section describes how you can create the report definition for profiles.

Follow these steps:

1. Edit the reports.xml file and include the report definition for the properties of the Profile table that you want to report on.
2. Edit the reports_schema.xml file and include the schema definition for the report. You can also generate the schema definition using a tool of your choice and include it in the reports_schema.xml file.

The following example shows the report definition to report on the Profile table:

```
<report>
    <Name>Profiles</Name>
        <Profile>
            <Report_Output>
                <Property>
                        <Type>ID</Type>
                        <Type>Name</Type>
                        <Type>Description</Type>
                        <Type>Is_BuiltIn</Type>
                </Property>
            </Report_Output>
        </Profile>
</report>
```

Column Name	Description
Id	Unique identifier for the profile
Name	Name of the profile
Description	Description associated with the profile
Is_BuiltIn	Integer "1" indicating if the profile is a built-in profile.

Report on Profile Assignments

This section describes how you can create the report definition for profile assignment.

Follow these steps:

1. Edit the reports.xml file and include the report definition for the properties of the Profile Assignment table that you want to report on.
2. Edit the reports_schema.xml file and include the schema definition for the report. You can also generate the schema definition using a tool of your choice and include it in the reports_schema.xml file.

The following example shows the report definition to report on the Profile Assignment table:

```
<report>
    <Name>Profile Assignments</Name>
        <Profile_Assignment>
            <Report_Output>
                <Property>
                        <Type>Catalog_ID</Type>
                        <Type>Profile_ID</Type>
                        <Type>User_ID</Type>
                </Property>
            </Report_Output>
        </Profile_Assignment>
</report>
```

Column Name	Description
Catalog_Id	Identifier for the catalog for which profile is assigned
Profile_Id	Identifier for the profile assigned
User_Id	Identifier for the User for which the profile is assigned

Report on Permissions

This section describes how you can create the report definition for permissions.

Follow these steps:

1. Edit the reports.xml file and include the report definition for the properties of the Permission table that you want to report on.
2. Edit the reports_schema.xml file and include the schema definition for the report. You can also generate the schema definition using a tool of your choice and include it in the reports_schema.xml file.

The following example shows the report definition to report on the Permission table:

```
<report>
    <Name>Permissions</Name>
        <Permission>
            <Report_Output>
                <Property>
                        <Type>ID</Type>
                        <Type>Action_Name</Type>
                        <Type>Securable_Name</Type>
                </Property>
            </Report_Output>
        </Permission>
</report>
```

Column Name	Description
Id	Identifier for the permission
Action_Name	Action associated with the permission
Securable_Name	Securable associated with the permission

Report on Permission Assignment

This section describes how you can create the report definition for permission assignments.

Follow these steps:

1. Edit the reports.xml file and include the report definition for the properties of the Permission Assignment table that you want to report on.
2. Edit the reports_schema.xml file and include the schema definition for the report. You can also generate the schema definition using a tool of your choice and include it in the reports_schema.xml file.

The following example shows the report definition to report on the Permission Assignment table:

```
<report>

    <Name>Permission Assignments</Name>

        <Permission_Assignment>

            <Report_Output>

                <Property>

                        <Type>Profile_ID</Type>

                        <Type>Permission_ID</Type>

                </Property>

             </Report_Output>

        </Permission_Assignment>

</report>
```

Column Name	Description
Profile_Id	Profile identifier for the profile
Permission_Id	Permission identifier for the permission assigned on the profile

Report on Actions

This section describes how to create the report definition for actions. The Action table stores the actions for a particular securable. For example, actions are View, Open, Save, and so on.

Follow these steps:

1. Edit the reports.xml file and include the report definition for the properties of the Action table that you want to report on.
2. Edit the reports_schema.xml file and include the schema definition for the report. You can also generate the schema definition using a tool of your choice and include it in the reports_schema.xml file.

The following example shows the report definition to report on the Action table:

```
<report>
    <Name>Actions</Name>
        <Action>
            <Report_Output>
               <Property>
                        <Type>Name</Type>
                        <Type>Display_Name</Type>
                </Property>
             </Report_Output>
        </Action>
</report>
```

Column Name	Description
Name	Action identifier
Display_Name	Display name for the action identifier

Report on Securables

This section describes how to create the report definition for securables. The Actions table stores the securables for a particular action. For example, actions are Mart, Model, Entity, User Management, and so on.

Follow these steps:

1. Edit the reports.xml file and include the report definition for the properties of the Securable table that you want to report on.
2. Edit the reports_schema.xml file and include the schema definition for the report. You can also generate the schema definition using a tool of your choice and include it in the reports_schema.xml file.

The following example shows the report definition to report on the Securable table.

```
<report>
    <Name>Actions</Name>
        <Securable>
            <Report_Output>
                <Property>
                        <Type>Name</Type>
                        <Type>Parent_Name</Type>
                        <Type>Display_Name</Type>
                </Property>
            </Report_Output>
        </Securable>
</report>
```

Column Name	Description
Name	Securable identifier
Parent_Name	Parent securable item if any
Display_Name	Display name for the securable identifier