Oracle® Cloud

SMML Schema Reference for Oracle Analytics Cloud

F38574-10
July 2024

Oracle Cloud SMML Schema Reference for Oracle Analytics Cloud,

F38574-10

Primary Author: Stefanie Rhone

Contributors: Oracle Analytics Cloud development, product management, and quality assurance teams

Contents

Preface

Learn how to use the service to explore and analyze data by creating workbooks and reports.

Topics:

Audience

This guide is intended for data modelers and business intelligence analysts who use Oracle Analytics Cloud:

- **Data Modelers** use the Semantic Modeler to create, design, edit, and deploy semantic models to Oracle Analytics Cloud.
- **Analysts** use the deployed semantic model's subject areas to model enterprise data and create workbooks, analyses, and dashboards for consumers. Analysts can select interactive visualizations and create advanced calculations to reveal data insights.

Documentation Accessibility

For information about Oracle's commitment to accessibility, visit the Oracle Accessibility Program website at http://www.oracle.com/pls/topic/lookup?ctx=acc&id=docacc.

Access to Oracle Support

Oracle customer access to and use of Oracle support services will be pursuant to the terms and conditions specified in their Oracle order for the applicable services.

Diversity and Inclusion

Oracle is fully committed to diversity and inclusion. Oracle respects and values having a diverse workforce that increases thought leadership and innovation. As part of our initiative to build a more inclusive culture that positively impacts our employees, customers, and partners, we are working to remove insensitive terms from our products and documentation. We are also mindful of the necessity to maintain compatibility with our customers' existing technologies and the need to ensure continuity of service as Oracle's offerings and industry standards evolve. Because of these technical constraints, our effort to remove insensitive terms is ongoing and will take time and external cooperation.

Related Documents

These related Oracle resources provide more information.

- For a full list of guides, refer to the Guides tab in the Oracle Analytics Cloud Help Center.

Conventions

Conventions used in this document are described in this topic.

Text Conventions

Convention	Meaning
boldface	Boldface type indicates graphical user interface elements associated with an action, or terms defined in text or the glossary.
italic	Italic type indicates book titles, emphasis, or placeholder variables for which you supply particular values.
`monospace`	Monospace type indicates commands within a paragraph, URLs, code in examples, text that appears on the screen, or text that you enter.

Videos and Images

Your company can use skins and styles to customize the look of the application, dashboards, reports, and other objects. It is possible that the videos and images included in the product documentation look different than the skins and styles your company uses.

Even if your skins and styles are different than those shown in the videos and images, the product behavior and techniques shown and demonstrated are the same.

1 About the Semantic Modeler Markup Language and Schema Properties

This guide describes the Semantic Modeler feature.

This chapter provides information about the Semantic Modeler Markup Language (SMML) and schema properties. SMML schema files correspond to objects in a semantic model. You can use SMML schema files for metadata migration, programmatic metadata generation and manipulation, metadata patching, and other functions.

Topics:

About the Semantic Modeler Markup Language (SMML)

The Semantic Modeler Markup Language (SMML) is a JSON-based markup language that describes an object and its elements in a SMML schema JSON file.

A SMML schema JSON file represents each object in a semantic model. A semantic model is comprised of a set of SMML schema files. You can use SMML schema files for metadata migration, programmatic metadata generation and manipulation, metadata patching, and other functions.

To view and edit SMML schema files in a semantic model, right-click an object and then select **Open in SMML Editor.** See Edit Semantic Model Objects Using the SMML Editor.

Required Properties

In SMML schema files, required properties are properties that must contain a value. For example, you must include the `name` property for any object and its elements.

Boolean Values

In a SMML schema file, elements with boolean values are `FALSE` by default unless you have configured the value to `TRUE` by editing the object or its SMML schema file in your semantic model.

About SMML Schema Files and Folders

SMML schema files are generated for each object in a semantic model and automatically sorted based on their object type into root folders for the corresponding layer that the object type belongs to.

When you're working with a semantic model, a SMML schema file is generated for any object added to the model. Each schema file defines an object and its properties. When you add an object to a semantic model, a SMML schema file is generated and designated to one of four root folders depending on the type of object the file represents.

A root folder exists for each semantic model layer: physical, logical, and presentation. The Variables folder exists for initialization blocks and associated variable elements.

If the semantic model's object type names contain unsupported characters, then the folder names won't match the object type names. For information on how folders and files are named, see Naming Conventions for SMML.

SMML Physical Layer Folder

The physical layer root folder contains database object folders at the top-level. Database folders contain schema folders or catalog folders containing one or more schema folders grouped together. Each schema folder contains physical tables and if created, physical alias tables.

SMML Logical Layer Folder

The logical layer root folder contains business model object folders at the top-level. Each business model folder contains logical tables.

SMML Presentation Layer Folder

The presentation layer root folder contains subject area object folders at the top-level. Each subject area folder contains presentation tables.

SMML Variables Folder

The Variables root folder contains an initialization blocks folder and a variables subfolder, both at the top-level. The initialization blocks folder contains initialization block objects. The variables subfolder contains variables that are associated with the initialization block objects.

SMML Naming Conventions

SMML schema files and attributes follow specific naming conventions to enable effective mapping of objects in a semantic model and the SMML files that reference them.

SMML Schema Files

When you create a semantic model and add an object, Oracle Analytics generates the SMML schema file a JSON file. The file name matches the name of the object it represents. For example, if a database object is named Sample Data, then the SMML schema file is also named Sample Data. The object name and SMML schema file names match except in the case when the object name contains unsupported characters. See Object Naming Criteria for SMML Schema Files and Folders.

Attribute Names

Attributes names referenced in SMML files follow lower camel case format. For example, `sourceType`.

Fully Qualified Names

Fully qualified names (FQNs) in a SMML schema file refer to objects from other SMML schema files. The format for a fully qualified name is the object's type, followed by a colon (`:`), followed by the fully qualified path of the object, and the name, separated by the period (`.`) character. In the SMML editor, the FQN displays as *`objecttype:fullyqualifiedpath.name`*. For example, `physicalcolumn:Sample App Data.SAMPLE. D02 Time Month Grain.Per_Name_Qtr`.

Note:

As the period (`.`) character is used as a separator, if an object's name contains a period (`.`) character then it is escaped by a backslash (`\`) character.

Object Naming Criteria for SMML Schema Files and Folders

You can name SMML schema files and folders by following the object-specific naming criteria for maximum length, spaces, unsupported characters, and unique naming rules.

Object Type	Unique Naming Rules	Maximum Length (characters)	Leading / Trailing Blank Spaces Allowed	Unsupported Characters
Business Model	None	128	No	Asterisk (`*`) Question mark (`?`) Single quote (`'`)
Catalog Schema	None	128	Yes	Asterisk (`*`) Question mark (`?`)
Connection Pool	None	128	Yes	Asterisk (`*`) Question mark (`?`)
Database	None	128	Yes	Asterisk (`*`) Question mark (`?`)
Initialization Blocks	None	128	Yes	Asterisk (`*`) Question mark (`?`)
Logical Table	None	128	No	Asterisk (`*`) Question mark (`?`) Single quote (`'`)
Logical Column Logical Level Logical Table Source	None	128	No	Asterisk (`*`) Question mark (`?`) Single quote (`'`)

Object Type	Unique Naming Rules	Maximum Length (characters)	Leading / Trailing Blank Spaces Allowed	Unsupported Characters
Physical Table Physical Table Alias	Use unique physical table and unique physical table alias names. You can't use the same names when the object shares a parent.	128	Yes	Asterisk (*) Question mark (?)
Presentation Table	Use unique presentation table names. You can't share the same name as the parent subject area.	128	No	Asterisk (*) Question mark (?) Single quote (')
Presentation Column Presentation Hierarchy Presentation Level	None	128	No	Asterisk (*) Question mark (?) Single quote (')
Subject Area	Use unique subject area names. You can't use the same name for any child tables.	128	No	Asterisk (*) Question mark (?) Single quote (')
Variable	Use unique variable names. You can't use the same name as any other variable associated with any initialization block in the semantic model.	128	Yes	Asterisk (*) Question mark (?)

Replacing Unsupported Characters in File and Folder Names

Certain characters in the names of your SMML schema files and folders are replaced automatically if they are unsupported for the object type of your file. The characters replaced differ based on which operating system you use.

Characters Replaced on Windows Systems

Characters in SMML schema file and folder names replaced on Windows systems if they are unsupported for an object type are the following:

Character	Replaced By Character If Unsupported
Double quotation mark (")	Underscore (_)
Asterisk (*)	Underscore (_)
Slash (/)	Underscore (_)
Colon (:)	Underscore (_)
Less-than symbol (<)	Underscore (_)
Greater-than symbol (>)	Underscore (_)
Question mark (?)	Underscore (_)

Character	Replaced By Character If Unsupported
Backslash (\)	Underscore (_)
Vertical bar (|)	Underscore (_)

Characters Replaced on Unix Systems

Characters in SMML schema file and folder names replaced on Unix systems if they are unsupported for an object type are the following:

Character	Replaced By Character If Unsupported
Slash (/)	Underscore (_)

Edit Semantic Model Objects Using the SMML Editor

You can use the SMML editor to view and edit the JSON SMML schema file of an object in your semantic model.

The SMML editor displays a semantic model object's text-based JSON SMML schema file based on the object-type JSON schema. If you are viewing or editing an invalid file, syntax and semantic errors are marked on the relevant line of text.

For more information about SMML, see SMML Schema Reference for Oracle Analytics Cloud.

1. On the Home page, click **Navigator** and then click **Semantic Models**.
2. In the Semantic Models page, click a semantic model to open it.
3. In the semantic model's left pane, select a layer.
4. Locate the object you want to edit.
5. Right-click the object and then select **Open in SMML Editor**.
6. Edit the SMML schema file and click **Save** to save the semantic model.

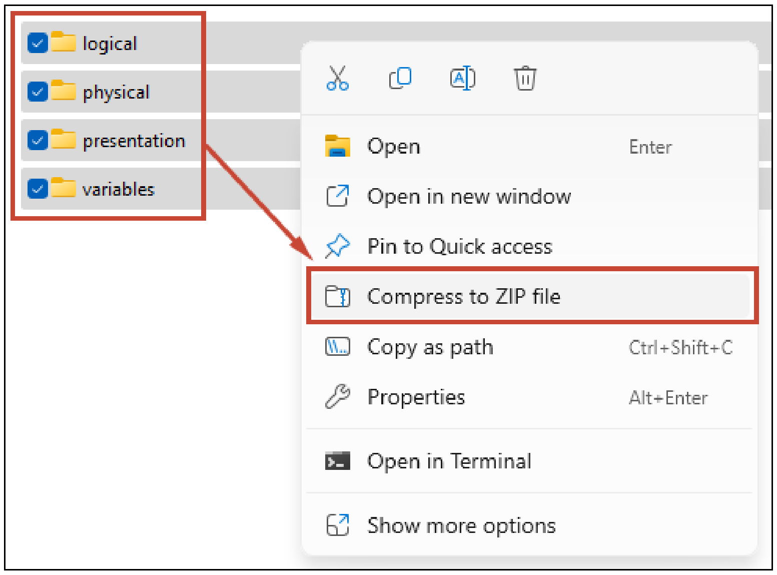
Compress SMML Files to a .zip File on Windows

If you use a development tool other than the SMML editor to create or edit a semantic model, you must compress the SMML files to a .zip file. You import this .zip file into Semantic Modeler to create or update a model in your Oracle Analytics instance.

Be sure that you don't compress the SMML files at the root folder. If you compress at the root folder and import the resulting .zip file into Semantic Modeler, the import fails.

Instead navigate into the root folder to select and compress the logical, physical, presentation, and variables folders.

1. In Windows, open File Explorer and navigate to the root folder containing your Semantic Model's SMML development files.
2. Double click the root folder to display the logical, physical, presentation, and variables child folders.
3. Select all child folders, right-click, and select **Compress to ZIP file**.



Use Terminal to Compress SMML Files to a .zip File on Mac

If you use a development tool other than the SMML editor to create or edit a semantic model, you must use the `zip` command to compress the SMML files to a .zip file. You import this .zip file into Semantic Modeler to create or update a model in your Oracle Analytics instance.

You must use the `zip` command to compress the SMML files to a .zip file. If you use the Compress menu option in Finder, the compress to ZIP fails.

Don't run the `zip` command on the root folder. If you compress the root folder and import the resulting .zip file into Semantic Modeler, the import fails.

Instead in the `zip` command, navigate to the root folder and specify the logical, physical, presentation, and variables folders.

1. Open Terminal and navigate to the root folder containing your Semantic Model's SMML development files. This folder contains the physical, logical, presentation and variables child folders.
2. Run this command to compress the folders into a .zip file:

```
zip -r <path>/<zip file name>.zip physical logical presentation variables
```

Defining Expressions Using SMML

You can use expressions in SMML schema files to define and identify joins between objects. These joins are referenced using the elements `expressionTemplate` and `expressionObjects`.

In SMML schema files, `expressionTemplate` defines and stores the join's expression, which can be viewed and edited using the Expression Editor. The `expressionObjects` element names the objects that are referenced in the join expression. For example:

```
  "physicalExpression": {
                        "expressionTemplate": " TIMESTAMPDIFF( SQL_TSI_DAY ,
%1, %2)",
                        "expressionObjects": [
                           "physicalColumn:Sample App Data (ORCL).SAMPLE.F13
Rev\\. (Order Dt Join).Order_Day_Dt",
                           "physicalColumn:Sample App Data (ORCL).SAMPLE.F13
Rev\\. (Order Dt Join).Ship_Day_Dt"
                        ]
                     }
```

For information on expression elements and the Expression Editor, see Expression Editor Reference in *Building Semantic Models in Oracle Analytics Cloud*.

2

SMML Physical Elements

This chapter provides SMML Schema reference information for physical elements. Physical elements correspond to objects in the physical layer of a semantic model.

Topics:

SMML Elements: Database

The SMML database element corresponds to the database schema which is part of the physical layer of a semantic model. The database schema contains database objects and elements.

Database Elements

- `name` (required property) — The name of the database.
- `description` — The description of the database.
- `tags` — The keywords assigned to this object. This element corresponds to the **Tags** field.
- `databaseType` (required property) — The type of data source. For example, Oracle Database, SQL Server, or DB2. See DataBase Type Enumerated Values.
- `persistConnectionPool` — References the connection pool used as the persist connection pool, if one has been assigned. A persist connection pool is a database property that's used for specific types of queries (typically used to support Marketing queries).
- `connectionPools` — References the connection pools for this database object.
- `featureOverrides` — Lists the SQL features for this database.
- `joins` — Defines the joins for this link. It contains different child elements, depending on the type of join.
- `queryLimits` — When selected, allows all users to run `POPULATE SQL`. If you want most, but not all, users to be able to run `POPULATE SQL`, select this option and then limit queries for specific users or groups. See Set Query Limits in *Building Semantic Models in Oracle Analytics Cloud*.
- `virtualPrivateDatabase` — If set to `TRUE`, identifies the physical database source as a virtual private database (VPD). When a VPD is used, returned data results are contingent on the user's authorization credentials. This option is used with the Security Sensitive option for session variables.
- `crmMetadataTables` — For legacy Siebel Systems sources only. If set to `TRUE`, Oracle Analytics looks for the table definition in Oracle's Siebel CRM-specific tables.

- `allowDirectDatabaseRequests` — If set to `TRUE`, allows all users to run physical queries. The Oracle Analytics query engine sends unprocessed, user-entered, physical SQL directly to an underlying database. If you want most, but not all, users to be able to run physical queries, select this option and then limit queries for specific users or groups. See Set Query Limits in *Building Semantic Models in Oracle Analytics Cloud*.
- `allowPopulateQueries` — If set to `TRUE`, allows all users to run `POPULATE SQL` . If you want most, but not all, users to be able to run `POPULATE SQL`, select this option and then limit queries for specific users or groups. See Set Query Limits in *Building Semantic Models in Oracle Analytics Cloud*.

ConnectionPool Elements

- `name` (required property) — The name of the connection.
- `description` — The description of the connection.
- `connection` (required property) — References the connection defined in Oracle Analytics Cloud.
- `callInterface` — The call interface type.
- `maxConnections`— The maximum number of total connections allowed to this connection pool for a given user.
- `requiresFullyQualifiedTableNames` — If set to `TRUE`, indicates that this database requires fully qualified table names. The fully qualified names are based on the physical object names in the semantic model.
- `connectionTimeOut` — The amount of time that a connection to the data source remains open after a request completes. During this time, new requests use this connection rather than open a new one (up to the number specified for the maximum connections). The time is reset after each completed connection request. If you set the timeout to 0 (the default), connection pooling is disabled. In other words, each connection to the data source terminates immediately when the request completes. Any new connections either use another connection pool or open a new connection.
- `connectionTimeoutUnit` — The unit of measure such as `MINUTES`.
- `multithreaded` — If set to `TRUE`, indicates that the connection pool supports multi-threading, or in other words, that one connection pool can support multiple queries. If this option is set to false, each query is tied to a single database connection.
- `supportParams` — Indicates whether the data source supports parameters. If set to `false` and the database features table supports parameters, a special code runs that enables the Oracle Analytics query engine to push filters (or calculations) with parameters to the database.
- `isolationLevel` — For ODBC and DB2 gateways. Sets the transaction isolation level on each connection to the back-end database. The isolation level setting controls the default transaction locking behavior for all statements issued by a connection. Options are as follows:
  - **`default`** uses the default transaction locking behavior of the data source.
  - **`readCommitted`** specifies that shared locks are held while the data is read to avoid dirty reads.
  - **`readUncommitted`** implements dirty read (isolation level 0 locking). When this option is set, it is possible to read uncommitted or dirty data, change values in the data, and have rows appear or disappear in the data set before the end of the transaction.

  - **`repeatableRead`** places locks on all data that is used in a query, preventing other users from updating the data.
  - **`serializable`** places a range lock on the data set, preventing other users from updating or inserting rows into the data set until the transaction is complete.
- `runOnConnectScripts` — Contains a connection script that is run before the connection is established.
- `runBeforeQueryScripts` — Contains a connection script that is run before the query is run.
- `runAfterQueryScripts` — Contains a connection script that is run after the query is run.
- `runOnDisconnectScripts` — Contains a connection script that is run after the connection is closed.
- `writeBackConfig` — Indicates the data source's write back properties.
- `permissions` — Lists the users and application roles and permissions to access the connection pool. See Permission Elements.

RunScript Elements

- `script` (required property) — Contains the connection script.
- `disable` — If set to `TRUE`, used to disable the connection script.

WriteBackConfig Elements

- `dbSupportsUnicode` — This attribute is set to `TRUE` when working with columns of an explicit Unicode data type, such as `NCHAR`, in an Unicode database.
- `bulkInsertBufferSize` — Used for limiting the amount of data in kilobytes, each time data is inserted into a database table.
- `transactionBoundary` — Controls the batch size for an insert in a database table.
- `tempTablePrefix` — The first two characters in a temporary table name. The default value is `TT`.
- `tempTableOwner` — The table owner name used to qualify a temporary table name in a SQL statement, for example to create the table, `owner.tablename`.
- `tempTableDatabase` — Database where the temporary table is created. This property applies only to IBM OS/390, because IBM OS/390 requires the database name qualifier to be part of the `CREATE TABLE` statement.
- `tempTableSpace` — Tablespace where the temporary table will be created. This property applies only to IBM OS/390, because IBM OS/390 requires the tablespace name qualifier to be part of the `CREATE TABLE` statement.

Feature Elements

- `name` (required property) — Indicates the name of the feature, such as `LEFT_OUTER_JOIN_SUPPORTED`.
- `value` (required property) — Indicates whether this feature is supported by the database, or provides the actual value of the feature (such as "`0`" for `MAX_COLUMNS_IN_SELECT`).

QueryLimit Elements

- `accessor` (required property) — Lists the application role to assign query limits to.

- `maxRowSetting` — Specifies the status of the maximum number of rows limit. Valid values are:
  - **Inherit** — Inherits the `maxRows` limit from the parent application role. If there is no limit to inherit, then no limit is enforced.
  - **Enable** — Limits the number of rows to the value specified in `maxRows`. If the number of rows exceeds the `maxRows` value, then the query is terminated.
  - **Disable** — Disables any limits set in `maxRows`.
  - **Warn** — Doesn't enforce limits set in `maxRows`, but logs any queries that exceed the set limit in the query log.
- `maxRows` — Indicates the maximum number of rows the accessor can retrieve from the database.
- `maxTimeSetting` — Specifies the status of the maximum time limit. Valid values are:
  - **Inherit** — Inherits the `maxTime` limit from the parent application role. If there is no limit to inherit, then no limit is enforced.
  - **Enable** — Limits the number of minutes to the value specified in `maxTime`.
  - **Disable** — Disables any limits set in `maxTime`.
  - **Warn** — Doesn't enforce limits set in `maxTime`, but logs any queries that exceed the set limit in the query log.
- `maxTime` — The maximum number of minutes queries can run on a database.
- `logicalQueryMaxTime` — The maximum time in seconds a logical query can run.
- `directDatabaseRequests` — When selected, allows all users to run physical queries. The Oracle Analytics query engine will send unprocessed, user-entered, physical SQL directly to an underlying database. If you want most, but not all, users to be able to run physical queries, select this option and then limit queries for specific users or groups.
- `restrictions` — Indicates restrictions in place for accessing the database.

Restriction Elements

- `type` — Indicates the type of restriction, whether the restriction is `ALLOW` or `DENY`.
- `day` — The day specified for a restriction.
- `from` — Indicates the restriction start time.
- `to` — Indicates the restriction end time.

Syntax

```
{
    "$schema": "http://json-schema.org/draft-04/schema#",
    "$ref": "#/definitions/database",
    "definitions": {
        "database": {
            "type": "object",
            "additionalProperties": false,
            "properties": {
                "name": {
                    "type": "string"
                },
                "description": {
                    "type": "string"
                },
                "tags": {
                    "type": "array",
                    "items": {
                        "type": "string"
                    }
                },
                "databaseType": {
                    "$ref": "common_schemas#/definitions/DatabaseType"
                },
                "persistConnectionPool": {
                    "type": "string"
                },
                "connectionPools": {
                    "type": "array",
                    "items": {
                        "$ref": "#/definitions/ConnectionPool"
                    }
                },
                "featureOverrides": {
                    "type": "array",
                    "items": {
                        "$ref": "#/definitions/Feature"
                    }
                },
                "queryLimits": {
                    "type": "array",
                    "items": {
                        "$ref": "#/definitions/QueryLimit"
                    }
                },
                "virtualPrivateDatabase": {
                    "type": "boolean"
                },
                "crmMetadataTables": {
                    "type": "boolean"
                },
                "allowDirectDatabaseRequests": {
                    "type": "boolean"
                },
                "allowPopulateQueries": {
                    "type": "boolean"
                }
            },
            "required": [
                "databaseType",
                "name"
            ],
            "title": "Database"
        },
        "ConnectionPool": {
            "type": "object",
            "additionalProperties": false,
            "properties": {
                "name": {
                    "type": "string"
                },
                "description": {
                    "type": "string"
                },
                "connection": {
                    "type": "string"
                },
                "remoteConnection": {
                    "type": "boolean"
                },
                "maxConnections": {
                    "type": "integer"
                },
                "requiresFullyQualifedTableNames": {
                    "type": "boolean"
                },
                "connectionTimeout": {
                    "type": "integer"
                },
                "connectionTimeoutUnit": {
                    "type": "string",
                    "enum": ["WHEN_QUERY_COMPLETES", "DAYS", "HOURS",
"MINUTES", "SECONDS", "NEVER"]
                },
                "multithreaded": {
                    "type": "boolean"
                },
                "supportParams": {
                    "type": "boolean"
                },
                "isolationLevel": {
                    "type": "string"
                },
                "runOnConnectScripts": {
                    "type": "array",
                    "items": {
                        "$ref": "#/definitions/RunScript"
                    }
                },
                "runBeforeQueryScripts": {
                    "type": "array",
                    "items": {
                        "$ref": "#/definitions/RunScript"
                    }
                },
                "runAfterQueryScripts": {
                    "type": "array",
                    "items": {
                        "$ref": "#/definitions/RunScript"
                    }
                },
                "runOnDisconnectScripts": {
                    "type": "array",
                    "items": {
                        "$ref": "#/definitions/RunScript"
```

```
            }
        },
        "writeBackConfig": {
            "$ref": "#/definitions/WriteBackConfig"
        },
        "permissions": {
            "type": "array",
            "items": {
                "$ref": "common_schemas#/definitions/Permission"
            }
        }
    },
    "required": [
        "connection",
        "name"
    ],
    "title": "ConnectionPool"
},
"RunScript": {
    "type": "object",
    "additionalProperties": false,
    "properties": {
        "script": {
            "type": "string"
        },
        "disable": {
            "type": "boolean"
        }
    },
    "required": [
        "script"
    ],
    "title": "RunScript"
},
"WriteBackConfig": {
    "type": "object",
    "additionalProperties": false,
    "properties": {
        "dbSupportsUnicode": {
            "type": "boolean"
        },
        "bulkInsertBufferSize": {
            "type": "integer"
        },
        "transactionBoundary": {
            "type": "integer"
        },
        "tempTablePrefix": {
            "type": "string"
        },
        "tempTableOwner": {
            "type": "string"
        },
        "tempTableDatabase": {
            "type": "string"
        },
                "tempTableSpace": {
                    "type": "string"
                }
            },
            "title": "WriteBackConfig"
        },
        "Feature": {
            "type": "object",
            "additionalProperties": false,
            "properties": {

                "name": {
                    "type": "string"
                },
                "value": {
                    "oneOf": [
                        {"type": "string"},
                        {"type": "boolean"},
                        {"type": "integer"}
                    ]
                }
            },
            "required": [
                "name",
                "value"
            ],
            "title": "Feature"
        },
        "QueryLimit": {
            "type": "object",
            "additionalProperties": false,
            "properties": {
                "accessor": {
                    "type": "string"
                },
                "maxRowSetting": {
                    "type": "string",
                    "enum": ["INHERIT", "ENABLE", "DISABLE", "WARN"]
                },
                "maxRows": {
                    "type": "integer"
                },
                "maxTimeSetting": {
                    "type": "string",
                    "enum": ["INHERIT", "ENABLE", "DISABLE", "WARN"]
                },
                "maxTime": {
                    "type": "integer"
                },
                "logicalQueryMaxTime": {
                    "type": "integer"
                },
                "directDatabaseRequests": {
                    "type": "string",
                    "enum": ["INHERIT", "ALLOW", "DISALLOW"]
                },
                "restrictions": {
                    "type": "array",
                    "items": {
                        "$ref": "#/definitions/Restriction"
                    }
                }
            },
            "required": [
                "accessor"
            ],
            "title": "QueryLimit"
        },
        "Restriction": {
            "type": "object",
            "additionalProperties": false,
            "properties": {
                "type": {
                    "type": "string",
                    "enum": ["ALLOW", "DENY"]
                },
                "day": {
                    "type": "string",
                    "enum": ["SUNDAY", "MONDAY", "TUESDAY", "WEDNESDAY", 
"THURSDAY", "FRIDAY", "SATURDAY"]
                },
                "from": {
                    "type": "integer"
                },
                "to": {
                    "type": "integer"
                }
            },
            "required": [
                "day",
                "from",
                "to",
                "type"
            ],
            "title": "Restriction"
        }
    }
}
```

SMML Elements: Catalog

The SMML catalog element corresponds to the catalog schema which is part of the physical layer. The catalog schema contains the catalog object and elements.

Catalog Elements

- `name` (required property) — The name of the catalog.
- `description` — The description of the catalog.
- `tags` — The keywords assigned to this object. This element corresponds to the **Tags** field.

- `dynamicName` — If a session variable is being used to specify the name of the catalog, this element references that session variable.

Syntax

```
{
    "$schema": "http://json-schema.org/draft-04/schema#",
    "$ref": "#/definitions/catalog",
    "definitions": {
        "catalog": {
            "type": "object",
            "additionalProperties": false,
            "properties": {
                "name": {
                    "type": "string"
                },
                "description": {
                    "type": "string"
                },
                "tags": {
                    "type": "array",
                    "items": {
                        "type": "string"
                    }
                },
                "dynamicName": {
                    "type": "string"
                }
            },
            "required": [
                "name"
            ],
            "title": "PhysicalCatalog"
        }
    }
}
```

SMML Elements: Schema

The SMML schema element corresponds to the schema object in the physical layer of a semantic model.

Schema Elements

- `name` (required property) — The name of the schema.
- `description` — The description of the schema.
- `tags` — The keywords assigned to this object. This element corresponds to the **Tags** field.
- `dynamicName` — If a session variable is being used to specify the name of this physical schema, this element references that session variable.

Syntax

```
{
    "$schema": "http://json-schema.org/draft-04/schema#",
    "$ref": "#/definitions/schema",
    "definitions": {
        "schema": {
            "type": "object",
            "additionalProperties": false,
            "properties": {
                "name": {
                    "type": "string"
                },
                "description": {
                    "type": "string"
                },
                "tags": {
                    "type": "array",
                    "items": {
                        "type": "string"
                    }
                },
                "dynamicName": {
                    "type": "string"
                }
            },
            "required": [
                "name"
            ],
            "title": "PhysicalSchema"
        }
    }
}
```

SMML Elements: Physical Table Alias

The SMML physical table alias element corresponds to the physical table alias object in the physical layer of a semantic model.

Physical Table Alias Elements

- `name` (required property) — The name of the physical table alias.
- `description` — The description of the physical table alias.
- `tags` — Keywords assigned to this object. This element corresponds to the **Tags** field.
- `sourceTable` (required property) — For alias tables only. References the physical table that is being used as the source for this alias table.
- `additionalKeys` — Keys that can be defined in addition to primary and display columns.
- `caching` — If set to `TRUE`, indicates that this table is included in the Oracle Analytics query engine query cache. See Caching Elements.
- `dynamicName` — If a session variable is being used to specify the name of this physical table alias, this element references that session variable.

- `overrideSourceCacheSetting` — For alias tables only. If set to `TRUE`, indicates that the alias table has its own cache properties that override the cache properties of the source table.

Syntax

```
{
    "$schema": "http://json-schema.org/draft-04/schema#",
    "$ref": "#/definitions/physicalTableAlias",
    "definitions": {
        "physicalTableAlias": {
            "type": "object",
            "additionalProperties": false,
            "properties": {
                "name": {
                    "type": "string"
                },
                "description": {
                    "type": "string"
                },
                "tags": {
                    "type": "array",
                    "items": {
                        "type": "string"
                    }
                },
                "sourceTable": {
                    "type": "string"
                },
                "additionalKeys": {
                    "type": "array",
                    "items": {
                        "type": "array",
                        "items": {
                            "type": "string"
                        }
                    }
                },
                "caching": {
                    "$ref": "common_schemas#/definitions/Caching"
                },
                "dynamicName": {
                    "type": "string"
                },
                "overrideSourceCacheSetting": {
                    "type": "boolean"
                }
            },
            "required": [
                "name",
                "sourceTable"
            ],
            "title": "PhysicalTableAlias"
        }
        }
}
```

SMML Elements: Physical Table

The SMML physical table element corresponds to the physical table schema which is part of the physical layer in a semantic model. The physical table schema contains physical table objects and elements.

Physical Table Elements

- `name` (required property) — The name of the physical table.
- `description` — The description of the physical table.
- `tags` — Keywords assigned to this object. This element corresponds to the **Tags** field.
- `sourceType` — Indicates the type of physical table.
- `sourceTable` — For alias tables only. References the physical table that is being used as the source for this alias table.
- `additionalKeys` — Keys that can be defined in addition to primary and display columns.
- `joins` — Defines the joins for this link. It contains different child elements, depending on the type of join.
- `dynamicName` — If a session variable is being used to specify the name of this physical table, this element references that session variable.
- `sqlHints` — For Oracle Databases only. Lists any database hints, such as index hints or leading hints, that were specified for this physical table.
- `overrideSourceCacheSetting` — For alias tables only. If set to `TRUE`, indicates that the alias table has its own cache properties that override the cache properties of the source table.
- `caching` — If set to true, indicates that this table is included in the Oracle Analytics query engine query cache. See Caching Elements.
- `eventPollingFrequency` — The polling frequency, in seconds. Only applies if this table is anOracle Analytics query engine event polling table. The default value is 3600 seconds.
- `selectStatements` — List of select statements that can be defined for different databases.
- `physicalColumns` — Specifies the physical columns that belong to this physical table.

SelectStatement Elements

- `databaseType` — The type of data source. See DataBase Type Enumerated Values.
- `query` — Data source specific query for the physical table.

PhysicalColumn Elements

- `name` (required property) — The name of the physical column.
- `description` — The description of the physical column.
- `dataType` — The data type of the physical column, such as `VARCHAR`. See DataType Enumerated Values.
- `length` — The length of the physical column.

- `nullable` — If set to `TRUE`, indicates that null values are allowed for the column. This allows null values to be returned to the user, which is expected with certain functions and with outer joins.

Join Elements

- `joinType` — Indicates the type of join. See JoinType Enumerated Values.
- `useJoinExpression` — If set to `TRUE`, then specify the join expression.
- `rightTable` (required property) — Indicates the join's right table.
- `cardinality` — Indicates the cardinality of the join. See Cardinality Enumerated Values.
- `hint` — For Oracle Databases only. Lists any database hints, such as index hints or leading hints, that were specified for this complex join.
- `joinConditions` — A list of matching columns in a join between two tables. A join condition defines a relationship between two tables. It involves columns that relate the two tables. A join condition may have more than one column in a table related to columns in another table.
- `joinExpression` — A join condition defined as an expression instead of simply matching columns from two tables. For example, `Table1.Column1 = Table2.Column1 AND Table1.Column2 = <value>`. See Expression Elements.

Cardinality Enumerated Values

- `ONE_TO_ONE`
- `ZERO_OR_ONE_TO_ONE`
- `ONE_TO_ZERO_OR_ONE`
- `ZERO_OR_ONE_TO_ZERO_OR_ONE`
- `ONE_TO_MANY`
- `ZERO_OR_ONE_TO_MANY`
- `MANY_TO_ONE`
- `MANY_TO_ZERO_OR_ONE`
- `MANY_TO_MANY`
- `UNKNOWN`

ComplexJoinCondition Elements

- `expressionTemplate` (required property) — Defines and stores the join's expression.
- `expressionObjects` — The objects referenced in the join expression.

JoinCondition Elements

- `leftColumn` (required property) — Indicates columns in a join condition from the left side of a table that is in a join.
- `rightColumn` (required property) — Indicates columns in a join condition from the right side of a table that is in a join.

Syntax

```
{
    "$schema": "http://json-schema.org/draft-04/schema#",
    "$ref": "#/definitions/physicalTable",
    "definitions": {
        "physicalTable": {
            "type": "object",
            "additionalProperties": false,
            "properties": {
                "name": {
                    "type": "string"
                },
                "description": {
                    "type": "string"
                },
                "tags": {
                    "type": "array",
                    "items": {
                        "type": "string"
                    }
                },
                "sourceType": {
                    "type": "string",
                    "enum": [
                        "TABLE",
                        "STORED_PROCEDURE",
                        "SELECT"
                    ]
                },
                "sourceTable": {
                    "type": "string"
                },
                "additionalKeys": {
                    "type": "array",
                    "items": {
                        "type": "array",
                        "items": {
                            "type": "string"
                        }
                    }
                },
                "joins": {
                  "type": "array",
                  "items": {
                      "$ref": "#/definitions/Join"
                  }
                },
                "dynamicName": {
                    "type": "string"
                },
                "sqlHints": {
                    "type": "string"
                },
                "caching": {
                    "$ref": "common_schemas#/definitions/Caching"
                },
                "overrideSourceCacheSetting": {
                    "type": "boolean"
                },
                "eventPollingFrequency": {
                    "type": "string"
                },
                "selectStatements": {
                    "type": "array",
                    "items": {
                        "$ref": "#/definitions/SelectStatement"
                    }
                },
                "physicalColumns": {
                    "type": "array",
                    "items": {
                        "$ref": "#/definitions/PhysicalColumn"
                    }
                }
            },
            "required": [
                "name"
            ],
            "title": "PhysicalTable"
        },
        "SelectStatement": {
            "type": "object",
            "additionalProperties": false,
            "properties": {
                "databaseType": {
                    "$ref": "common_schemas#/definitions/DatabaseType"
                },
                "query": {
                    "type": "string"
                }
            },
            "title": "SelectStatement"
        },
        "PhysicalColumn": {
            "type": "object",
            "additionalProperties": false,
            "properties": {
                "name": {
                    "type": "string"
                },
                "description": {
                    "type": "string"
                },
                "dataType": {
                    "$ref": "common_schemas#/definitions/DataType"
                },
                "length": {
                    "type": "integer"
                },
                "nullable": {
```

```
                "type": "boolean"
            }
        },
        "required": [
            "name"
        ],
        "title": "PhysicalColumn"
    },
    "Join": {
        "type": "object",
        "additionalProperties": false,
        "properties": {
           "joinType": {
              "$ref": "common_schemas#/definitions/JoinType"
           },
           "useJoinExpression": {
              "type": "boolean"
           },
           "rightTable": {
              "type": "string",
              "rawType": "table"
           },
           "cardinality": {
              "$ref": "#/definitions/Cardinality"
           },
           "hint": {
              "type": "string"
           },
           "joinConditions": {
              "type": "array",
              "items": {
                    "$ref": "#/definitions/JoinCondition"
              }
           },
           "joinExpression": {
              "$ref": "common_schemas#/definitions/Expression"
           }

        },
        "required": [
           "rightTable"
        ],
        "title": "Join"
    },
    "Cardinality": {
        "type": "string",
        "enum": [
           "ONE_TO_ONE",
           "ZERO_OR_ONE_TO_ONE",
           "ONE_TO_ZERO_OR_ONE",
           "ZERO_OR_ONE_TO_ZERO_OR_ONE",
           "ONE_TO_MANY",
           "ZERO_OR_ONE_TO_MANY",
           "MANY_TO_ONE",
           "MANY_TO_ZERO_OR_ONE",
           "MANY_TO_MANY",
                  "UNKNOWN"
               ],
               "title": "Cardinality"
           },
           "ComplexJoinCondition": {
               "type": "object",
               "additionalProperties": false,
               "properties": {
                  "expressionTemplate": {
                     "type": "string"
                  },
                  "expressionObjects": {
                     "type": "array",
                     "items": {
                           "type": "string"
                     }
                  }
               },
               "required": [
                  "expressionTemplate"
               ],
               "title": "ComplexJoinCondition"
            },
            "JoinCondition": {
               "type": "object",
               "additionalProperties": false,
               "properties": {
                  "leftColumn": {
                     "type": "string",
                     "rawType": "column"
                  },
                  "rightColumn": {
                     "type": "string",
                     "rawType": "column"
                  }
               },
               "required": [
                  "leftColumn",
                  "rightColumn"
               ],
               "title": "JoinCondition"
            }
        }
    }
```

3

SMML Logical Elements

This chapter provides SMML Schema reference information for logical elements. Logical elements typically correspond to objects in the logical layer of a semantic model.

Topics:

- SMML Elements: Business Model
- SMML Elements: Logical Table

SMML Elements: Business Model

The SMML business model element corresponds to the business model schema that is part of the logical layer in a semantic model. The business model schema contains business model objects and their elements.

Business Model Elements

- `name` (required property) — The name of the business model.
- `description` — The description of the business model.
- `tags` — The keywords assigned to this object. This element corresponds to the **Tags** field.
- `disable` — If set to `TRUE`, used to disable the logical table.

Syntax

```
{
    "$schema": "http://json-schema.org/draft-04/schema#",
    "$ref": "#/definitions/businessModel",
    "definitions": {
        "businessModel": {
            "type": "object",
            "additionalProperties": false,
            "properties": {
                "name": {
                    "type": "string"
                },
                "description": {
                    "type": "string"
                },
                "tags": {
                    "type": "array",
                    "items": {
                        "type": "string"
                    }
                },
                "disable": {
                    "type": "boolean"
                }
            },
            "required": [
                "name"
            ],
            "title": "BusinessModel"
        }
    }
}
```

SMML Elements: Logical Table

The SMML logical table element corresponds to the logical table schema in the logical layer of a semantic model. The logical table schema contains logical objects and elements.

logicalTable Elements

- `name` (required property) — The name of the logical table.
- `description` — The description of the logical table.
- `tags` — Keywords assigned to this object. This element corresponds to the **Tags** field.
- `type` (required property) — Indicates the type of table.
- `primaryKey` — Indicates that the key is the logical primary key for the table.
- `logicalColumns` — Lists the logical column objects in the logical layer.
- `logicalTableSources` — References the logical table sources for this logical table.
- `joins` — Defines the joins for this link. It contains different child elements, depending on the type of join.
- `hierarchyType` — Indicates the type of hierarchy.
- `levelBasedHierarchy` — Indicates whether this is a level-based hierarchy.
- `parentChildHierarchy` — Indicates whether this is a parent-child hierarchy.
- `dataFilters` — Data filters defined for this object.

LevelBasedHierarchy Elements

- `defaultRootLevel` — References the default root level of this dimension. When there are many root levels in a dimension, the default root level is the one that's used for drill-down.
- `ragged` — If set to true, indicates that this dimension hierarchy is unbalanced. An unbalanced hierarchy is one that contains levels that are not at the same depth.
- `skipped` — If set to true, indicates that this dimension hierarchy is a skip-level hierarchy. A skip-level hierarchy is one in which there are members that do not have a value for a particular parent level.
- `logicalLevels` (required property) — References the logical levels for this dimension.
- `logicalHierarchies` (required property) — References the logical hierarchies for this dimension.

ParentChildHierarchy Elements

- `name` (required property) — The name of this parent child hierarchy.
- `description` — The description of this parent child hierarchy.

- `logicalLevels` (required property) — References the logical levels for this dimension
- `relationshipTables` (required property) — References the relationship table defined for the parent-child hierarchy.

LogicalHierarchy Elements

- `name` (required property) — The name of this logical hierarchy.
- `description` — The description of this logical hierarchy.
- `levels` (required property) — References the child levels that have been defined for this logical level.

LogicalLevel Elements

- `name` (required property) — The name of this logical level.
- `grandTotalLevel` — If set to `TRUE`, indicates that this level is the Grand Total level, which is a special level representing the grand total for a dimension. Each dimension can have just one Grand Total level. A Grand Total level doesn't contain dimensional attributes and doesn't have a level key.
- `displayKey` — This element references the columns designated to be used for display for this hierarchy.
- `preferredDrillPath` — If a drill path has been defined that's outside the normal drill path defined by the dimension level hierarchy, this element references the level users should drill to.
- `primaryKey` — Indicates that this key is the logical primary key for the table.
- `additionalKeys` — Keys that can be defined in addition to primary and display columns.
- `parentKey` — References the parent key for this level (for dimensions with parent-child hierarchy only).
- `chronologicalKey` — Indicates that this key is a chronological key for a time dimension.
- `disableAggregateToHigherLevel` — By default, measure values at a particular level constitute aggregated measures. Set to `TRUE` to disable.
- `numberofElements` — The number of elements that exist at this logical level.

LogicalColumn Elements

- `name` (required property) — The name of this logical column.
- `description` — The description of this logical column.
- `dataType` — The datatype of the logical column. See DataType Enumerated Values.
- `sortBy` — If the sort order for this logical column is based on a different logical column, this element indicates the name of the logical column used for sorting.
- `descriptorColumn` — When multilingual columns are based on a lookup function, it's common to specify the non-translated lookup key column as the descriptor ID column of the translated column. This element references the descriptor ID column.
- `writeable` — When set to `TRUE`, indicates that write-back has been enabled for this column. This feature is typically used with ADF Business Component data sources.
- `logicalColumnSource` (required property) — References the source for this logical column.
- `aggregation` — Aggregation function applied to the data in the column.

- `dataFilters` — Data filters defined for this object. See DataFilter Elements.
- `logicalLevel` — For dimension columns, the level this column has been assigned to. For level-based measures, the level at which the column has been explicitly fixed.

LogicalColumnSource Elements

- `derivedFrom` (required property) — References the logical table source that this column mapping belongs to. Each column mapping must specify only one logical table source.
- `physicalMappings` — Contains the expression that identifies the physical column for this column mapping. See Expression Elements.
- `logicalExpression` — Contains the expression that identifies the logical column for this column mapping. See Expression Elements.

PhysicalMapping Elements

- `logicalTableSource` (required property) — References the logical table source for this logical table.
- `physicalExpression` (required property) — Contains the expression that identifies the physical column for this column mapping. See Expression Elements.

LogicalTableSource Elements

- `name` (required property) — The name of the logical table source.
- `description` — The description of the logical table source.
- `disable` — If set to `TRUE`, used to disable the logical table source.
- `priority` — Indicates the priority group number of this logical table source. Logical table source priority group numbers are assigned to indicate which logical table source is used for queries where there is more than one logical table source that can satisfy the requested set of columns.
- `tableMapping` (required property) — Element that defines the mapping of logical table source to physical tables and the join types between the physical tables.
- `dataGranularity` — The level of detail of the data in the logical table source.
- `dataFragmentation` — Contains the expression that defines how the source is fragmented. See Expression Elements.
- `combineWithOtherFragments` — If set to `TRUE`, used to combine the data with other fragmented sources.
- `enableFragmentSelection` — If set to `TRUE`, used to enable the data driven fragment selection.
- `dataFilter` — Filter to limit rows for or in a logical table source. See DataFilter Elements.
- `distinctValues` — If set to `TRUE`, used if the values for the logical table source are unique.

TableMapping Elements

- `tables` — Physical tables mapped to a logical table source.
- `logicalTableSourceJoins` — Joins between tables in a logical table source.

LogicalTableSourceJoin Elements

- `leftTable` (required property) — Indicates the join's left table.

- `rightTable` (required property) — Indicates the join's right table.
- `joinType` — Indicates the type of join. See JoinType Enumerated Values.
- `disable` — If set to `TRUE`, disables the join between the physical tables.

Join Elements

- `rightTable` (required property) — Indicates the join's right table.
- `joinType` — Indicates the type of join. See JoinType Enumerated Values.
- `cardinality` — Indicates the cardinality of the join. See Cardinality Enumerated Values.
- `drivingTable` — If a driving table has been specified for the key, this attribute references that logical table.

Cardinality Enumerated Values

- `ONE_TO_ONE`
- `ZERO_OR_ONE_TO_ONE`
- `ONE_TO_ZERO_OR_ONE`
- `ZERO_OR_ONE_TO_ZERO_OR_ONE`
- `ONE_TO_MANY`
- `ZERO_OR_ONE_TO_MANY`
- `MANY_TO_ONE`
- `MANY_TO_ZERO_OR_ONE`
- `MANY_TO_MANY`
- `UNKNOWN`

Aggregation Elements

- `rule` — Aggregation rule for the column.
- `dataIsDense` — If set to `TRUE`, indicates that all sources to which this column is mapped have a row for every combination of logical hierarchy levels that they represent. Setting this option to `TRUE` when any table source used by this column doesn't contain dense data returns incorrect results.
- `countDistinctOverrides` — If the aggregation rule is `COUNT_DISTINCT`, you can specify a different aggregation rule for each logical table sources mapped to a logical column.
- `dimensionBasedRules` — Dimension-specific aggregation rules.
- `aggregateByLevels` — Level-based measure calculated to a specific level of aggregation.

DimensionBasedRule Elements

- `dimension` (required property) — Specifies the dimension for which an aggregation rule needs to be applied.
- `rule`(required property) — Aggregation rule for the column.
- `aggregateExpression` — If the rule is `EXPRESSION`, specify an aggregate expression for the dimension. See Expression Elements.

CountDistinctOverrides Elements

- `logicalTableSource` (required property) — References the logical table source.
- `rule` — Aggregation rule for the column.

DerivedFrom Enumerated Values

- `PHYSICAL_COLUMNS`
- `LOGICAL_COLUMNS`

AggregationRule Enumerated Values

- `NONE`
- `SUM`
- `AVG`
- `COUNT`
- `COUNT_DISTINCT`
- `MAX`
- `FIRST`
- `LAST`
- `MEDIAN`
- `STD_DEV`
- `STD_DEV_POP`
- `EVALUATE_AGGR`
- `BASED_ON_DIMENSION`

AggregationRuleBasedOnDimension Enumerated Values

- `NONE`
- `SUM`
- `AVG`
- `COUNT`
- `COUNT_DISTINCT`
- `MAX`
- `MIN`
- `FIRST`
- `LAST`
- `MEDIAN`
- `STD_DEV`
- `STD_DEV_POP`
- `EVALUATE_AGGR`

- `EXPRESSION`

RelationshipTable Elements

- `logicalTableSource` (required property) — References the logical table source for this logical table.
- `table` (required property) — Indicates the name of the parent-child relationship table that the source is based on.
- `memberKey` (required property) — The name of the column in the parent-child relationship table that identifies the member.
- `parentKey` (required property) — References the parent key for this level (for dimensions with parent-child hierarchies only).
- `distance` (required property) — The name of the column in the parent-child relationship table that specifies the number of parent-child hierarchical levels from the member to the ancestor.
- `leafNodeIdentifier` (required property) — The name of the column in the parent-child relationship table that indicates if the member is a leaf member.

Syntax

```
{
    "$schema": "http://json-schema.org/draft-04/schema#",
    "$ref": "#/definitions/logicalTable",
    "definitions": {
        "logicalTable": {
            "type": "object",
            "additionalProperties": false,
            "properties": {
                "name": {
                    "type": "string"
                },
                "description": {
                    "type": "string"
                },
                "tags": {
                    "type": "array",
                    "items": {
                        "type": "string"
                    }
                },
                "type": {
                    "type": "string",
                    "enum": [
                        "DIMENSION",
                        "FACT",
                        "LOOKUP"
                    ]
                },
                "primaryKey": {
                    "type": "array",
                    "items": {
                        "type": "string"
                    }
                },
                "logicalColumns": {
                    "type": "array",
                    "items": {
                        "$ref": "#/definitions/LogicalColumn"
                    }
                },
                "logicalTableSources": {
                    "type": "array",
                    "items": {
                        "$ref": "#/definitions/LogicalTableSource"
                    }
                },
                "joins": {
                    "type": "array",
                    "items": {
                        "$ref": "#/definitions/Join"
                    }
                },
                "hierarchyType": {
                    "type": "string"
                },
                "levelBasedHierarchy": {
                    "$ref": "#/definitions/LevelBasedHierarchy"
                },
                "parentChildHierarchy": {
                    "$ref": "#/definitions/ParentChildHierarchy"
                },
                "dataFilters": {
                    "type": "array",
                    "items": {
                        "$ref": "common_schemas#/definitions/DataFilter"
                    }
                }
            },
            "required": [
                "name",
                "type"
            ],
            "title": "LogicalTable"
        },
        "LevelBasedHierarchy": {
            "type": "object",
            "additionalProperties": false,
            "properties": {
                "defaultRootLevel": {
                    "type": "string"
                },
                "ragged": {
                    "type": "boolean"
                },
                "skipped": {
                    "type": "boolean"
                },
                "logicalLevels": {
                    "type": "array",
            "items": {
                "$ref": "#/definitions/LogicalLevel"
            }
        },
        "logicalHierarchies": {
            "type": "array",
            "items": {
                "$ref": "#/definitions/LogicalHierarchy"
            }
        }
    },
    "required": [
        "logicalHierarchies",
        "logicalLevels"
    ],
    "title": "LevelBasedHierarchy"
},
"ParentChildHierarchy": {
    "type": "object",
    "additionalProperties": false,
    "properties": {
        "name": {
            "type": "string"
        },
        "description": {
            "type": "string"
        },
        "logicalLevels": {
            "type": "array",
            "items": {
                "$ref": "#/definitions/LogicalLevel"
            }
        },
        "relationshipTables": {
            "type": "array",
            "items": {
                "$ref": "#/definitions/RelationshipTable"
            }
        }
    },
    "required": [
        "logicalLevels",
        "name",
        "relationshipTables"
    ],
    "title": "ParentChildHierarchy"
},
"LogicalHierarchy": {
    "type": "object",
    "additionalProperties": false,
    "properties": {
        "name": {
            "type": "string"
        },
        "description": {
            "type": "string"
        },
        "levels": {
            "type": "array",
            "items": {
                "type": "string"
            }
        }
    },
    "required": [
        "levels",
        "name"
    ],
    "title": "LogicalHierarchy"
},
"LogicalLevel": {
    "type": "object",
    "additionalProperties": false,
    "properties": {
        "name": {
            "type": "string"
        },
        "grandTotalLevel": {
            "type": "boolean"
        },
        "displayKey": {
            "type": "string"
        },
        "preferredDrillPath": {
            "type": "array",
            "items": {
                "type": "string"
            }
        },
        "primaryKey": {
            "type": "array",
            "items": {
                "type": "string"
            }
        },
        "additionalKeys": {
            "type": "array",
            "items": {
                "type": "array",
                "items": {
                   "type": "string"
                }
            }
        },
        "parentKey": {
            "type": "string"
        },
        "chronologicalKey": {
          "type": "array"
        },
        "disableAggregateToHigherLevel": {
            "type": "boolean"
```

```
                },
                "numberOfElements": {
                    "type": "integer"
                }
            },
            "required": [
                "name"
            ],
            "title": "LogicalLevel"
        },
        "LogicalColumn": {
            "type": "object",
            "additionalProperties": false,
            "properties": {
                "name": {
                    "type": "string"
                },
                "description": {
                    "type": "string"
                },
                "dataType": {
                    "$ref": "common_schemas#/definitions/DataType"
                },
                "sortBy": {
                    "type": "string"
                },
                "descriptorColumn": {
                    "type": "string"
                },
                "writeable": {
                    "type": "boolean"
                },
                "logicalColumnSource": {
                    "$ref": "#/definitions/LogicalColumnSource"
                },
                "aggregation": {
                    "$ref": "#/definitions/Aggregation"
                },
                "dataFilters": {
                    "type": "array",
                    "items": {
                        "$ref": "common_schemas#/definitions/DataFilter"
                    }
                },
                "logicalLevel": {
                    "type": "string"
                }
            },
            "required": [
                "logicalColumnSource",
                "name"
            ],
            "title": "LogicalColumn"
        },
        "LogicalColumnSource": {
            "type": "object",
            "additionalProperties": false,
            "properties": {
                "derivedFrom": {
                    "type": "string",
                    "enum": [
                        "PHYSICAL_COLUMNS",
                        "LOGICAL_COLUMNS"
                    ]
                },
                "physicalMappings": {
                    "type": "array",
                    "items": {
                        "$ref": "#/definitions/PhysicalMapping"
                    }
                },
                "logicalExpression": {
                    "$ref": "common_schemas#/definitions/Expression"
                }
            },
            "required": [
                "derivedFrom"
            ],
            "title": "LogicalColumnSource"
        },
        "PhysicalMapping": {
            "type": "object",
            "additionalProperties": false,
            "properties": {
                "logicalTableSource": {
                    "type": "string"
                },
                "physicalExpression": {
                    "$ref": "common_schemas#/definitions/Expression"
                }
            },
            "required": [
                "logicalTableSource",
                "physicalExpression"
            ],
            "title": "PhysicalMapping"
        },
        "LogicalTableSource": {
            "type": "object",
            "additionalProperties": false,
            "properties": {
                "name": {
                    "type": "string"
                },
                "description": {
                    "type": "string"
                },
                "disable": {
                    "type": "boolean"
                },
                "priority": {
                    "type": "integer"
                },
                "tableMapping": {
                    "$ref": "#/definitions/TableMapping"
                },
                "dataGranularity": {
                    "type": "array",
                    "items": {
                        "type": "string"
                    }
                },
                "dataFragmentation": {
                    "$ref": "common_schemas#/definitions/Expression"
                },
                "combineWithOtherFragments": {
                    "type": "boolean"
                },
                "enableFragmentSelection": {
                    "type": "boolean"
                },
                "dataFilter": {
                    "$ref": "common_schemas#/definitions/Expression"
                },
                "distinctValues": {
                    "type": "boolean"
                }
            },
            "required": [
                "name",
                "tableMapping"
            ],
            "title": "LogicalTableSource"
        },
        "TableMapping": {
            "type": "object",
            "additionalProperties": false,
            "properties": {
                "tables": {
                    "type": "array",
                    "items": {
                        "type": "string"
                    }
                },
                "logicalTableSourceJoins": {
                    "type": "array",
                    "items": {
                        "$ref": "#/definitions/LogicalTableSourceJoin"
                    }
                }
            },
            "title": "TableMapping"
        },
        "LogicalTableSourceJoin": {
            "type": "object",
            "additionalProperties": false,
            "properties": {
                "leftTable": {
                    "type": "string"
                },
                "rightTable": {
                    "type": "string"
                },
                "joinType": {
                    "$ref": "common_schemas#/definitions/JoinType"
                },
                "disable": {
                    "type": "boolean"
                }
            },
            "required": [
                "leftTable",
                "rightTable"
            ],
            "title": "LogicalTableSourceJoin"
        },
        "Join": {
            "type": "object",
            "additionalProperties": false,
            "properties": {
                "rightTable": {
                    "type": "string",
                    "rawType": "table"
                },
                "joinType": {
                    "$ref": "common_schemas#/definitions/JoinType"
                },
                "cardinality": {
                    "$ref": "#/definitions/Cardinality"
                },
                "drivingTable": {
                    "type": "string",
                    "rawType": "table"
                }
            },
            "required": [
                "rightTable"
            ],
            "title": "Join"
        },
        "Cardinality": {
            "type": "string",
            "enum": [
                "ONE_TO_ONE",
                "ZERO_OR_ONE_TO_ONE",
                "ONE_TO_ZERO_OR_ONE",
                "ZERO_OR_ONE_TO_ZERO_OR_ONE",
                "ONE_TO_MANY",
                "ZERO_OR_ONE_TO_MANY",
                "MANY_TO_ONE",
                "MANY_TO_ZERO_OR_ONE",
                "MANY_TO_MANY",
                "UNKNOWN"
             ],
            "title": "Cardinality"
        },
        "Aggregation": {
            "type": "object",
            "additionalProperties": false,
            "properties": {
                "rule": {
                    "$ref": "#/definitions/AggregationRule"
                },
                "dimensionBasedRules": {
                    "type": "array",
                    "items": {
                        "$ref": "#/definitions/DimensionBasedRule"
                    }
                },
                "dataIsDense": {
                    "type": "boolean"
                },
                "countDistinctOverrides": {
                     "type": "array",
                      "items": {
                        "$ref": "#/definitions/CountDistinctOverrides"
                    }
                },
                "aggregateByLevels": {
                    "type": "array",
                    "items": {
                        "type": "string"
                    }
                }
            },
            "title": "Aggregation"
        },
        "DimensionBasedRule": {
            "type": "object",
            "additionalProperties": false,
            "properties": {
                "dimension": {
                    "type": "string"
                },
                "rule": {
                    "$ref": "#/definitions/AggregationRuleBasedOnDimension"
                },
                "aggregateExpression": {
                    "$ref": "common_schemas#/definitions/Expression"
                }
            },
            "required": [
                "dimension",
                "rule"
            ],
            "title": "DimensionBasedRule"
        },
        "CountDistinctOverrides": {
            "type": "object",
            "additionalProperties": false,
            "properties": {
                "logicalTableSource": {
                    "type": "string"
                },
                "rule": {
                    "$ref": "#/definitions/AggregationRule"
                }
            },
            "title": "CountDistinctOverrides"
        },
        "DerivedFrom": {
            "type": "string",
            "enum": [
                "PHYSICAL_COLUMNS",
                "LOGICAL_COLUMNS"
            ],
            "title": "DerivedFrom"
        },
        "AggregationRule": {
            "type": "string",
            "enum": [
                "NONE",
                "SUM",
                "AVG",
                "COUNT",
                "COUNT_DISTINCT",
                "MAX",
                "MIN",
                "FIRST",
                "LAST",
                "MEDIAN",
                "STD_DEV",
                "STD_DEV_POP",
                "EVALUATE_AGGR",
                "BASED_ON_DIMENSION"
            ],
            "title":"AggregationRule"
        },
        "AggregationRuleBasedOnDimension": {
            "type": "string",
            "enum": [
                "NONE",
                "SUM",
                "AVG",
                "COUNT",
                "COUNT_DISTINCT",
                "MAX",
                "MIN",
                "FIRST",
                "LAST",
                "MEDIAN",
                "STD_DEV",
                "STD_DEV_POP",
                "EVALUATE_AGGR",
                "EXPRESSION"
            ],
            "title":"AggregationRule"
        },
         "RelationshipTable": {
            "type": "object",
            "additionalProperties": false,
            "properties": {
                "logicalTableSource": {
                    "type": "string"
                },
                "table": {
                    "type": "string"
                },
                "memberKey": {
                    "type": "string"
                },
                "parentKey": {
                    "type": "string"
                },
                "distance": {
                    "type": "string"
                },
                "leafNodeIdentifier": {
                    "type": "string"
                }
            },
            "required": [
                "distance",
                "leafNodeIdentifier",
                "logicalTableSource",
                "memberKey",
                "parentKey",
                "table"
            ],
            "title": "RelationshipTable"
        }
    }

}
```

4 SMML Presentation Elements

This chapter provides SMML reference information for presentation elements. Presentation elements correspond to objects in the presentation layer of a semantic model.

Topics:

SMML Elements: Subject Area

The SMML subject area element corresponds to the subject area schema in the presentation layer of a semantic model. The subject area schema contains subject area objects and elements.

subjectArea Elements

- `name` (required property) — The name of the subject area.
- `description` — The description of the subject area.
- `tags` — The keywords assigned to this object. This element corresponds to the **Tags** field.
- `sourceBusinessModel` (required property) — References the business model for this subject area.
- `implicitFactColumn` — References the implicit fact column for this subject area, if one has been set. This column is used to specify a default join path between dimension tables when there are several possible alternatives or contexts.
- `alternateNames` — Alternate names assigned to this object. This element corresponds to the **Alternate Names** field.
- `hideIfTrue` — An expression that controls whether this subject area is visible in Oracle Analytics. See Expression Elements.
- `tableOrder` — References the order of the tables that belong to the subject area.
- `permissions` — Permissions defined for this object. See Permission Elements.
- `localization` — Localized names for presentation layer subject areas and their descriptions.

TableOrder Elements

- `name` (required property) — The name of the presentation table.
- `children` — List of tables nested under the presentation table.

Children Elements

- `name` (required property) — The name of the child presentation table.
- `children` — List of tables nested under the table.

Syntax

```
{
    "$schema": "http://json-schema.org/draft-04/schema#",
    "$ref": "#/definitions/subjectArea",
    "definitions": {
        "subjectArea": {
            "type": "object",
            "additionalProperties": false,
            "properties": {
                "name": {
                    "type": "string"
                },
                "description": {
                    "type": "string"
                },
                "tags": {
                    "type": "array",
                    "items": {
                        "type": "string"
                    }
                },
                "sourceBusinessModel": {
                    "type": "string",
                    "rawType": "businessModel"
                },
                "implicitFactColumn": {
                    "type": "string",
                    "rawType": "logicalColumn"
                },
                "alternateNames": {
                    "type": "array",
                    "items": {
                        "type": "string"
                    }
                },
                "hideIfTrue": {
                    "$ref": "common_schemas#/definitions/Expression"
                },
                "tableOrder": {
                    "type": "array",
                    "items": {
                        "$ref": "#/definitions/TableOrder"
                    }
                },
                "permissions": {
                    "type": "array",
                    "items": {
                        "$ref": "common_schemas#/definitions/Permission"
                    }
                },
                "localization": {
                    "$ref": "common_schemas#/definitions/Localization"
                }
            },
            "required": [
                "name",
                "sourceBusinessModel"
            ],
            "title": "SubjectArea"
        },
        "TableOrder": {
            "type": "object",
            "additionalProperties": false,
            "properties": {
                "name": {
                    "type": "string"
                },
                "children": {
                    "type": "array",
                    "items": {
                        "$ref": "#/definitions/Child"
                    }
                }
            },
            "required": [
                "name"
            ],
            "title": "TableOrder"
        },
        "Child": {
            "type": "object",
            "additionalProperties": false,
            "properties": {
                "name": {
                    "type": "string"
                },
                "children": {
                    "type": "array",
                    "items": {
                        "$ref": "#/definitions/Child"
                    }
                }
            },
            "required": [
                "name"
            ],
            "title": "Child"
        }
    }
}
```

SMML Elements: Presentation Table

The SMML presentation table element corresponds to the presentation table schema in the presentation layer of a semantic model. The presentation table schema contains presentation table objects and their elements.

Presentation Table Elements

- `name` (required property) — The name of the presentation table.
- `description` — The description of the presentation table.
- `tags` — Keywords assigned to this object. This element corresponds to the **Tags** field.
- `alternateNames` — Alternate names assigned to this object. This element corresponds to the **Alternate Names** field.
- `hideIfTrue` — Contains the expression specified to control the visibility of this object. This element corresponds to the **Hide If** check box. See Expression Elements.
- `presentationColumns` — References the presentation columns that belong to this presentation table.
- `hierarchies` — References the hierarchies for this presentation table.
- `permissions` — Permissions defined for this object. See Permission Elements.
- `dataFilters` — Data filters defined for this object. See DataFilter Elements.
- `localization` — Localized names for presentation layer tables and their descriptions. See Localization Elements.

PresentationColumn Elements

- `name` (required property) — The name of this presentation column.
- `description` — The description of the presentation column.
- `tags` — Keywords assigned to this object. This element corresponds to the **Tags** field.
- `sourceLogicalColumn` — References the logical column sources.
- `alternateNames` — Alternate names assigned to this object. This element corresponds to the **Alternate Names** field.
- `hideIfTrue` — Contains the expression specified to control the visibility of this object. This element corresponds to the **Hide If** check box. See Expression Elements.
- `permissions` — Permissions defined for this object. See Permission Elements.
- `dataFilters` — Data filters defined for this object. See DataFilter Elements.
- `localization` — Localized names for presentation layer columns and their descriptions. See Localization Elements.

Hierarchy Elements

- `name` (required property) — The name of this hierarchy.
- `description` — The description of this hierarchy.
- `tags` — Keywords assigned to this object. This element corresponds to the **Tags** field.

- `displayColumn` — This element references the columns designated to be used for display for this hierarchy.
- `sourceLogicalTable` — References the logical table sources.
- `alternateNames` — Alternate names assigned to this object. This element corresponds to the **Alternate Names** field.
- `hideIfTrue` — Contains the expression specified to control the visibility of this object. This element corresponds to the **Hide If** check box. See Expression Elements.
- `permissions` — Permissions defined for this object. See Permission Elements.
- `dataFilters` — Data filters defined for this object. See DataFilter Elements.
- `localization` — Localized names for presentation layer hierarchies and their descriptions. See Localization Elements.
- `levels` — References the child levels that have been defined.

HierarchyLevel Elements

- `name` (required property) — The name of this hierarchy level.
- `description` — The description of this hierarchy level.
- `tags` — Keywords assigned to this object. This element corresponds to the **Tags** field.
- `sourceLogicalLevel` — References the logical levels.
- `displayColumn` — This element references the columns designated to be used for display for this hierarchy.
- `alternateNames` — Alternate names assigned to this object. This element corresponds to the **Alternate Names** field.
- `permissions` — Permissions defined for this object. See Permission Elements.
- `datafilters` — Data filters defined for this object. See DataFilter Elements.
- `localization` — Localized names for presentation layer hierarchies and their descriptions. See Localization Elements.

Syntax

```
{
    "$schema": "http://json-schema.org/draft-04/schema#",
    "$ref": "#/definitions/presentationTable",
    "definitions": {
        "presentationTable": {
            "type": "object",
            "additionalProperties": false,
            "properties": {
                "name": {
                    "type": "string"
                },
                "description": {
                    "type": "string"
                },
                "tags": {
                    "type": "array",
                    "items": {
                        "type": "string"
                        }
                    },
                    "alternateNames": {
                        "type": "array",
                        "items": {
                            "type": "string"
                        }
                    },
                    "hideIfTrue": {
                        "$ref": "common_schemas#/definitions/Expression"
                    },
                    "presentationColumns": {
                        "type": "array",
                        "items": {
                            "$ref": "#/definitions/PresentationColumn"
                        }
                    },
                    "hierarchies": {
                        "type": "array",
                        "items": {
                            "$ref": "#/definitions/Hierarchy"
                        }
                    },
                    "permissions": {
                        "type": "array",
                        "items": {
                            "$ref": "common_schemas#/definitions/Permission"
                        }
                    },
                    "dataFilters": {
                        "type": "array",
                        "items": {
                            "$ref": "common_schemas#/definitions/DataFilter"
                        }
                    },
                    "localization": {
                        "$ref": "common_schemas#/definitions/Localization"
                    }
                },
                "required": [
                    "name"
                ],
                "title": "PresentationTable"
            },
            "PresentationColumn": {
                "type": "object",
                "additionalProperties": false,
                "properties": {
                    "name": {
                        "type": "string"
                    },
                    "description": {
                        "type": "string"
                    },
                    "tags": {
                        "type": "array",
                "items": {
                    "type": "string"
                }
            },
            "sourceLogicalColumn": {
                "type": "string"
            },
            "alternateNames": {
                "type": "array",
                "items": {
                    "type": "string"
                }
            },
            "hideIfTrue": {
                "$ref": "common_schemas#/definitions/Expression"
            },
            "permissions": {
                "type": "array",
                "items": {
                    "$ref": "common_schemas#/definitions/Permission"
                }
            },
            "dataFilters": {
                "type": "array",
                "items": {
                    "$ref": "common_schemas#/definitions/DataFilter"
                }
            },
            "localization": {
                "$ref": "common_schemas#/definitions/Localization"
            }
        },
        "required": [
            "name"
        ],
        "title": "PresentationColumn"
    },
    "Hierarchy": {
        "type": "object",
        "additionalProperties": false,
        "properties": {
            "name": {
                "type": "string"
            },
            "description": {
                "type": "string"
            },
            "tags": {
                "type": "array",
                "items": {
                    "type": "string"
                }
            },
            "displayColumn": {
                "type": "string"
            },
            "sourceLogicalTable": {
                "type": "string"
            },
            "alternateNames": {
                "type": "array",
                "items": {
                    "type": "string"
                }
            },
            "hideIfTrue": {
                "$ref": "common_schemas#/definitions/Expression"
            },
            "permissions": {
                "type": "array",
                "items": {
                    "$ref": "common_schemas#/definitions/Permission"
                }
            },
            "dataFilters": {
                "type": "array",
                "items": {
                    "$ref": "common_schemas#/definitions/DataFilter"
                }
            },
            "localization": {
                "$ref": "common_schemas#/definitions/Localization"
            },
            "levels": {
                "type": "array",
                "items": {
                    "$ref": "#/definitions/HierarchyLevel"
                }
            }
        },
        "required": [
            "name"
        ],
        "title": "Hierarchy"
    },
    "HierarchyLevel": {
        "type": "object",
        "additionalProperties": false,
        "properties": {
            "name": {
                "type": "string"
            },
            "description": {
                "type": "string"
            },
            "tags": {
                "type": "array",
                "items": {
                    "type": "string"
                }
            },
            "sourceLogicalLevel": {
                "type": "string"
            },
            "displayColumn": {
                "type": "string"
            },
            "alternateNames": {
                "type": "array",
                "items": {
                    "type": "string"
                }
            },
            "permissions": {
                "type": "array",
                "items": {
                    "$ref": "common_schemas#/definitions/Permission"
                }
            },
            "dataFilters": {
                "type": "array",
                "items": {
                    "$ref": "common_schemas#/definitions/DataFilter"
                }
            },
            "localization": {
                "$ref": "common_schemas#/definitions/Localization"
            }
        },
        "required": [
            "name"
        ],
        "title": "HierarchyLevel"
    }

}
}
```

5

SMML Variable Elements

This chapter provides SMML reference information for variable elements in a semantic model.

Topic:

- SMML Elements: Initialization Blocks

SMML Elements: Initialization Blocks

The SMML initialization blocks element corresponds to the initialization block schema in the variables layer. The initialization block schema contains initialization block objects and elements in a semantic model.

initBlock Elements

- `name` (required property) — The name of the initialization block.
- `description` — The description of the initialization block.
- `tags` — Keywords assigned to this object. This element corresponds to the **Tags** field.
- `type` — The type of semantic model variable.
- `connectionPool` — References the connection pool for this initialization block. This element is only used if the data source type for this initialization block is Database or XML.
- `runSchedule` — For global variables, you can specify the day, date, and time for the start date, as well as a refresh interval.
- `disable` — If set to `TRUE`, indicates that this initialization block is disabled.
- `queryReturnsVariableNamesandValues` — If set to `TRUE`, session variables are created dynamically and their values are set when a session begins.
- `allowDeferredExecution` — If set to true, indicates that deferred execution of this initialization block is enabled. Deferred execution is used to speed up the server startup time, by preventing the execution of the SQL for the initialization block when the server starts. The SQL is issued and the variables are initialized only when one of the variables is used.
- `cacheQueryResult` — If set to `TRUE`, the results of the select statement for `queryReturnsVariableNamesandValues` init block will be cached.
- `selectStatements` — List of select statements that can be defined for different databases.
- `variables` — References the variables that are associated with this init block.
- `dependencies` — The dependent init blocks run before this init block.

SelectStatement Elements

- `databaseType` — The type of data source. See DataBase Type Enumerated Values.
- `query` — Data source-specific query for the physical table.

RunSchedule Elements

- `interval` — The refresh interval for this initialization block, in seconds. For semantic model initialization blocks only. The default value is 0.
- `intervalUnit` — Unit of time. For example, seconds, minutes, hours, days.
- `startingOn` — The day or the time when the schedule starts.

Variable Elements

- `name` (required property) — The name of the variable.
- `description` — The description of the variable.
- `value` — The default value.
- `enableUsersToSetValue` — If set to `TRUE`, lets session variables be set after the initialization block has populated the value (at user login) by calling the ODBC store procedure `NQSSetSessionValue()`. For example, this option lets non-administrators set this variable for sampling.
- `securitySensitive` — If set to `TRUE`, identifies the variable as sensitive to security for virtual private databases (VPDs). When filtering cache table matches, the Oracle Analytics query engine looks at the parent database of each column or table that is referenced in the logical request projection list. If the physical database source is a VPD, the Oracle Analytics query engine matches a list of security-sensitive variables to each prospective cache hit. Cache hits would only occur on cache entries that included and matched all security-sensitive variables.

Syntax

```
{
    "$schema": "http://json-schema.org/draft-04/schema#",
    "$ref": "#/definitions/initBlock",
    "definitions": {
        "initBlock": {
            "type": "object",
            "additionalProperties": false,
            "properties": {
                "name": {
                    "type": "string"
                },
                "description": {
                    "type": "string"
                },
                "tags": {
                    "type": "array",
                    "items": {
                        "type": "string"
                    }
                },
                "type": {
                    "type": "string",
                    "enum": [
                        "STATIC",
                        "GLOBAL",
                        "SESSION"
                ]
            },
            "connectionPool": {
                "type": "string"
            },
            "runSchedule": {
                "$ref": "#/definitions/RunSchedule"
            },
            "disable": {
                "type": "boolean"
            },
            "queryReturnsVariableNamesAndValues": {
                "type": "boolean"
            },
            "allowDeferredExecution": {
                "type": "boolean"
            },
            "cacheQueryResult": {
                "type": "boolean"
            },
            "selectStatements": {
                "type": "array",
                "items": {
                    "$ref": "#/definitions/SelectStatement"
                }
            },
            "variables": {
                "type": "array",
                "items": {
                    "$ref": "#/definitions/Variable"
                }
            },
            "dependencies": {
                "type": "array",
                "items": {
                    "type": "string"
                }
            }
        },
        "required": [
            "name"
        ],
        "title": "InitBlock"
    },
    "SelectStatement": {
        "type": "object",
        "additionalProperties": false,
        "properties": {
            "databaseType": {
                "$ref": "common_schemas#/definitions/DatabaseType"
            },
            "query": {
                "type": "string"
            }
        },
        "title": "SelectStatement"
        },
        "RunSchedule": {
            "type": "object",
            "additionalProperties": false,
            "properties": {
                "interval": {
                    "type": "integer"
                },
                "intervalUnit": {
                    "type": "string"
                },
                "startingOn": {
                    "type": "string"
                }
            },
            "title": "RunSchedule"
        },
        "Variable": {
            "type": "object",
            "additionalProperties": false,
            "properties": {
                "name": {
                    "type": "string"
                },
                "description": {
                    "type": "string"
                },
                "value": {
                    "type": "string"
                },
                "enableUsersToSetValue": {
                    "type": "boolean"
                },
                "securitySensitive": {
                    "type": "boolean"
                }
            },
            "required": [
                "name"
            ],
            "title": "Variable"
        }
    }
}
```

6
SMML Common Elements

This chapter provides SMML reference information for common elements in a semantic model.

Topic:

- SMML Elements: Common

SMML Elements: Common

Expression Elements

- `expressionTemplate` (required property) — Defines and stores the join's expression.
- `expressionObjects` — The objects referenced in the join expression.

DataType Enumerated Values

- `BINARY`
- `BIT`
- `BOOLEAN`
- `CHAR`
- `DATETIME`
- `DOUBLE`
- `FLOAT`
- `NUMBER`
- `INT`
- `INTERVAL`
- `LONGVARBINARY`
- `LONGVARCHAR`
- `NUMERIC`
- `OBJECT`
- `SMALLINT`
- `SMALUINT`
- `TIME`
- `TIMESTAMP`
- `TINYINT`
- `TINYUINT`
- `UINT`

- `UNKNOWN`
- `VARBINARY`
- `VARCHAR`

Permission Elements

- `accessor` (required property) — An application role to assign access.
- `access` (required property) — The permission access value. The value can be `READ`, `WRITE`, and `NO ACCESS`.

DataFilter Elements

- `accessor` (required property) — Lists the application role to assign query limits to.
- `functionalGroup` — Functional groups defined for the application roles with different data access filters on the same semantic model object, usually a logical fact table.
- `status` — Specifies if the data filter is enabled.
- `filter` (required property) — The criteria for the data filter.

Localization Elements

- `localizationKey` — A tag assigned to the presentation object for localization translation.
- `nameVariable` — The variable that provides the value for the localization display name.
- `descriptionVariable` — The variable that provides the value for the localization description.

DataBaseType Enumerated Values

- `ORACLE_12C`
- `SQL_SERVER_2012`
- `TERADATA_V13`
- `INFORMIX_IDS_9`
- `DB2_AS400`
- `SYBASE_ASE_15`
- `SYBASE_IQ_12`
- `MYSQL`
- `IMPALA`
- `APACHE_SPARK`
- `REDSHIFT`
- `MONGO_DB`
- `SNOWFLAKE`
- `MONETDB`
- `DEFAULT`

Caching Elements

- `enable` — If set to `TRUE`, data is cached.
- `expiryTime` — The time duration after which a cache entry expires.
- `expiryUnit` — Unit of time. For example, `SECONDS, MINUTES, HOURS, DAYS`.

JoinType Enumerated Values

- `INNER`
- `LEFT_OUTER`
- `RIGHT_OUTER`
- `FULL_OUTER`
- `FULL_OUTER)STITCH`

Syntax

```
{
    "$schema": "http://json-schema.org/draft-04/schema#",
    "type": "object",
    "additionalProperties": false,
    "definitions": {
        "Expression": {
            "type": "object",
            "additionalProperties": false,
            "properties": {
                "expressionTemplate": {
                    "type": "string"
                },
                "expressionObjects": {
                    "type": "array",
                    "items": {
                        "type": "string"
                    }
                }
            },
            "required": [
                "expressionTemplate"
            ],
            "title": "Expression"
        },
        "DataType": {
            "type": "string",
            "enum": [
                "BINARY",
                "BIT",
                "BOOLEAN",
                "CHAR",
                "DATE",
                "DATETIME",
                "DOUBLE",
                "FLOAT",
                "NUMBER",
                "INT",
                "INTERVAL",
                "LONGVARBINARY",
                "LONGVARCHAR",
                "NUMERIC",
                "OBJECT",
                "SMALLINT",
                "SMALUINT",
                "TIME",
                "TIMESTAMP",
                "TINYINT",
                "TINYUINT",
                "UINT",
                "UNKNOWN",
                "VARBINARY",
                "VARCHAR"
            ],
            "title": "DataType"
        },
        "Permission": {
            "type": "object",
            "additionalProperties": false,
            "properties": {
                "accessor": {
                    "type": "string"
                },
                "access": {
                    "type": "string",
                    "enum": ["WRITE", "READ", "NO_ACCESS"]
                }
            },
            "required": [
                "access",
                "accessor"
            ],
            "title": "Permission"
        },
        "DataFilter": {
            "type": "object",
            "additionalProperties": false,
            "properties": {
                "accessor": {
                    "type": "string"
                },
                "functionalGroup": {
                    "type": "string"
                },
                "status": {
                    "type": "string",
                    "enum": ["ENABLED", "DISABLED", "IGNORED"]
                },
                "filter": {
                    "$ref": "#/definitions/Expression"
                }
            },
            "required": [
                "accessor",
                "filter"
            ],
            "title": "DataFilter"
        },
        "Localization": {
            "type": "object",
            "additionalProperties": false,
            "properties": {
                "localizationKey": {
                    "type": "string"
                },
                "nameVariable": {
                    "type": "string"
                },
                "descriptionVariable": {
                    "type": "string"
                }
            },
            "title": "Localization"
        },
        "DatabaseType": {
            "type": "string",
            "enum": [
                "ORACLE_DATABASE",
                "ORACLE_ADW",
                "SQL_SERVER",
                "TERADATA",
                "INFORMIX",
                "DB2",
                "SYBASE_ASE",
                "SYBASE_IQ",
                "MYSQL",
                "IMPALA",
                "APACHE_SPARK",
                "REDSHIFT",
                "MONGO_DB",
                "SNOWFLAKE",
                "MONETDB",
                "DEFAULT"
            ],
            "title": "DatabaseType"
        },
        "Caching": {
            "type": "object",
            "additionalProperties": false,
            "properties": {
                "enable": {
                    "type": "boolean"
                },
                "expiryTime": {
                    "type": "integer"
                },
                "expiryUnit": {
                    "enum": [
                        "SECONDS",
                        "MINUTES",
                        "HOURS",
                        "DAYS"
                    ]
                }
            },
            "title": "Caching"
        },
        "JoinType": {
            "type": "string",
            "enum": [
                "INNER",
                "LEFT_OUTER",
                "RIGHT_OUTER",
                "FULL_OUTER",
                "FULL_OUTER_STITCH"
            ],
            "title": "JoinType"
        }
    }
}
```