This guide provides information on using APIs to interface with Oracle LSH and Oracle DMW.
Preface ............................................................................................................................................................... xv

Audience ....................................................................................................................................................... xv
Documentation Accessibility ................................................................................................................. xv
Finding Information and Patches on My Oracle Support ........................................................................ xv
Finding Oracle Documentation ............................................................................................................ xvii
Related Documents ................................................................................................................................. xviii
Conventions .................................................................................................................................................. xviii

Part I Essential Information

1 Using Application Programming Interfaces
1.1 About APIs ............................................................................................................................................. 1-1
1.2 Calling APIs from Outside the Oracle Life Sciences Data Hub ............................................................... 1-3
1.2.1 Security Setup Required ................................................................................................................... 1-3
1.2.2 Calling the Security API Package ...................................................................................................... 1-3
1.2.3 Calling APIs from SAS ...................................................................................................................... 1-4
1.2.4 Using a Permanent Schema for Deploying Programs that Call APIs ............................................. 1-4
1.3 Calling APIs from Defined Programs .................................................................................................... 1-4
1.4 Code Example Using Security and Error Message APIs ....................................................................... 1-5

2 Reference Information
2.1 CDR Naming Version Object Type ......................................................................................................... 2-1
2.2 CDR Base Object Type ............................................................................................................................ 2-3
2.3 CDR Object-Specific Database Object Types .......................................................................................... 2-3
2.4 Retrieving Reference Codelist Names and Values .................................................................................. 2-4
2.5 Retrieving the Instance Domain ID ......................................................................................................... 2-4
2.6 Standard Parameters .................................................................................................................................. 2-5

Part II Object APIs

3 Application Areas
3.1 Define and Modify Application Areas ..................................................................................................... 3-1
3.1.1 Create an Application Area ................................................................................................................ 3-1
3.1.2 Modify an Application Area ................................................................................................................ 3-2
4 Business Areas

4.1 Define and Modify Business Areas ................................................................. 4-1
  4.1.1 Create a Business Area ............................................................................. 4-1
  4.1.2 Modify a Business Area ........................................................................... 4-3
  4.1.3 Check Out a Business Area ....................................................................... 4-3
  4.1.4 Undo a Business Area Checkout ................................................................. 4-4
  4.1.5 Check In a Business Area ......................................................................... 4-5
  4.1.6 Remove a Business Area ......................................................................... 4-5
  4.2 Create and Modify Business Area Hierarchies .............................................. 4-6
    4.2.1 Create a Business Area Hierarchy ......................................................... 4-6
    4.2.2 Modify a Hierarchy and a Hierarchy Column ......................................... 4-6
    4.2.3 Reorder a Hierarchy Column ................................................................. 4-7
    4.2.4 Remove a Hierarchy Column ................................................................. 4-8
    4.2.5 Remove a Business Area Hierarchy ...................................................... 4-8
  4.3 Create and Modify Business Area Joins ...................................................... 4-9
    4.3.1 Create a Join .......................................................................................... 4-9
    4.3.2 Modify a Join and a Join Column ......................................................... 4-10
    4.3.3 Remove a Join Column .......................................................................... 4-11
    4.3.4 Remove a Join ...................................................................................... 4-11
  4.4 Public APIs and Naming Views for Generic Visualization Adapter (GVA) ...... 4-12
    4.4.1 Initialize a Generic Visualization Business Area Instance ................... 4-12
    4.4.2 Reset a Generic Visualization Business Area ....................................... 4-13
    4.4.3 Get Possible Blinding Types of a Business Area Instance ................... 4-13
    4.4.4 Get Snapshot Labels Common to all Tables in a BA Instance for a Given Blinding AccessType 4-13
    4.4.5 Naming Views ........................................................................................ 4-14

5 Data Marts

5.1 Define and Modify Work Areas ..................................................................... 5-1
  5.1.1 Create a Data Mart .................................................................................. 5-1
  5.1.2 Check In a DataMart Definition ............................................................... 5-3
  5.1.3 Modify a Data Mart ................................................................................ 5-3
  5.1.4 Check Out a Data Mart .......................................................................... 5-4
  5.1.5 Remove a Data Mart .............................................................................. 5-4

6 Domains

6.1 Define and Modify Domains ....................................................................... 6-1
  6.1.1 Create a Domain .................................................................................... 6-1
  6.1.2 Modify a Domain ................................................................................... 6-2
  6.1.3 Copy a Domain ..................................................................................... 6-2
  6.1.4 Move Objects into a Domain ................................................................... 6-3
  6.1.5 Copy Objects into a Domain ................................................................... 6-4
11.1.2 Add and Modify an Entry ................................................................. 11-3
11.1.3 Copy a Report Set Entry into Another .............................................. 11-4
11.1.4 Modify a Report Set Entry ............................................................... 11-5
11.1.5 Move a Report Set Entry into Another .............................................. 11-5
11.1.6 Reorder Report Set Entries in a Parent Report Set ......................... 11-6
11.1.7 Find if a Report Set is Checked Out ................................................ 11-7
11.1.8 Check Unique and Strict Numbering in a Report Set ....................... 11-7
11.1.9 Identify if a Report Set Contains Child Entries ................................. 11-8
11.1.10 Find if a User has Modify Permission ............................................ 11-8
11.1.11 Remove an Object from a Report Set Entry .................................... 11-9
11.1.12 Remove a Report Set Entry ............................................................ 11-10
11.1.13 Get a Report Set Name ................................................................. 11-10
11.1.14 Get a Title .................................................................................. 11-11
11.1.15 Get a Chapter Number ................................................................. 11-11
11.1.16 Get a Parent Number .................................................................. 11-12
11.1.17 Get a List of Report Set Entry Titles ............................................. 11-12
11.1.18 Get All RSE Titles in a Report Set ............................................... 11-13
11.1.19 Get Attribute Values Derived from a Parent ................................. 11-13
11.1.20 Get the Lowest Entry Number ..................................................... 11-14
11.1.21 Get the Total Number of Report Set Entries ................................. 11-15
11.1.22 Create a Narrative ..................................................................... 11-15
11.1.23 Update a Narrative .................................................................. 11-16
11.1.24 Delete a Narrative .................................................................. 11-16
11.1.25 Check if Copying Retains Valid Numbering in a Target Report Set .. 11-17
11.1.26 Check if a Move Retains Valid Numbering in a Target Report Set .... 11-18
11.1.27 Check if a Move Retains Valid Numbering in the Parent Report Set .. 11-18
11.1.28 Check if Removal Retains Valid Numbering in a Parent Report Set ... 11-19
11.1.29 Check if Reordering Retains Valid Numbering in a Parent Report Set 11-19
11.1.30 Unassign a Planned Output ........................................................... 11-20
11.2 Create and Modify Report Sets .......................................................... 11-20
11.2.1 Create a Report Set ................................................................. 11-21
11.2.2 Check Out a Report Set ............................................................... 11-22
11.2.3 Undo a Report Set Checkout ......................................................... 11-23
11.2.4 Copy Objects Into a Report Set ................................................... 11-23
11.2.5 Get a Summary Output Validation Status ...................................... 11-24
11.2.6 Modify a Report Set ................................................................. 11-25
11.2.7 Move Objects into a Report Set ................................................... 11-26
11.2.8 Remove Objects from a Report Set .............................................. 11-27
11.2.9 Check In a Report Set ............................................................... 11-27
11.2.10 Remove a Report Set Definition ................................................ 11-28
11.2.11 Remove a Report Set ................................................................. 11-28
11.3 Create and Modify Overlay Template Definitions ............................ 11-29
11.3.1 Create an Overlay Template Definition .................................... 11-29
11.3.2 Modify an Overlay Template Definition File Definition ................. 11-30
11.3.3 Get an Overlay Template Definition File as a BLOB ....................... 11-30
11.3.4 Remove an Overlay Template Definition File Definition ................ 11-31
11.4 Report Set Overlay Template .......................................................... 11-31
12  Software Source Codes
12.1  Create and Modify Source Code................................................................. 12-1
12.1.1  Create a Source Code Object................................................................. 12-1
12.1.2  Get a Source Code CLOB........................................................ ............... 12-3
12.1.3  Modify Source Code................................................................................ 12-3
12.1.4  Set the Primary Flag to Yes................................................................. 12-5
12.1.5  Update a Shareable Flag....................................................................... 12-5
12.1.6  Remove a Source Code Object............................................................ 12-6

13  Tables
13.1  Create and Modify Tables........................................................................ 13-1
13.1.1  Create a Table Definition.................................................................... 13-1
13.1.2  Create a Table Instance....................................................................... 13-2
13.1.3  Create a Temporary Blob..................................................................... 13-4
13.1.4  Create a Table Column....................................................................... 13-4
13.1.5  Create a Table Constraint.................................................................... 13-5
13.1.6  Modify a Table Definition.................................................................... 13-6
13.1.7  Modify a Table Descriptor.................................................................... 13-7
13.1.8  Modify a Table Instance....................................................................... 13-7
13.1.9  Reorder a Column................................................................................ 13-8
13.1.10 Upload a Table Descriptor or Column.................................................. 13-9
13.1.11 Check In a Table Object...................................................................... 13-9
13.1.12 Remove a Single Object....................................................................... 13-10

14  Parameters
14.1  Define and Modify Parameters................................................................ 14-1
14.1.1  Create a Parameter.............................................................................. 14-1
14.1.2  Check Out a Parameter......................................................................... 14-3
14.1.3  Check In a Parameter.......................................................................... 14-3
14.1.4  Get Displayed Parameter Values.......................................................... 14-4
14.2  Define Parameter Relations....................................................................... 14-5
14.2.1  Create a Parameter Relation Collection.............................................. 14-5
14.2.2  Get Parameter Instances for Value Passing........................................ 14-6
14.2.3  Remove Parameter Relations.............................................................. 14-7

15  Variables
15.1  Create and Modify Variables.................................................................... 15-1
Part III  Common APIs

19  Setup Utilities

19.1  Initialize APIs ................................................................. 19-1
  19.1.1  Initialize a Package ......................................................... 19-1
  19.1.2  Verify Whether an API is Enabled ...................................... 19-2
  19.1.3  Enable an API ............................................................... 19-2
  19.1.4  Disable an API ............................................................... 19-3
  19.2  Define and Modify Adapters .............................................. 19-3
  19.2.1  Create an Adapter Domain ............................................. 19-3
  19.2.2  Modify an Adapter Domain ............................................ 19-4
  19.2.3  Create an Adapter Area .................................................. 19-5
  19.2.4  Modify an Adapter Area .................................................. 19-5
  19.2.5  Populate a Tech Type Table .......................................... 19-6
  19.2.6  Modify a Tech Type Table .............................................. 19-7
  19.3  Host Definition Constants ................................................ 19-7
  19.4  Get Factory Support .......................................................... 19-7
  19.4.1  Get a Naming Version Object ......................................... 19-8
  19.4.2  Get a User ID ............................................................... 19-8
  19.4.3  Get a User Name ........................................................... 19-9
  19.5  Get Factory Utilities .......................................................... 19-9
  19.5.1  Get a Base Object Type .................................................. 19-9
  19.5.2  Get a Company ID ......................................................... 19-10
  19.6  Get Factory Validations ....................................................... 19-10
  19.6.1  Validate a Namespace .................................................... 19-10
  19.6.2  Validate a Reference ...................................................... 19-11
  19.7  Get Data from Naming Tables ........................................... 19-11
  19.7.1  Get the Latest Version .................................................... 19-12
  19.7.2  Get a Maximum Version ................................................ 19-12
  19.7.3  Get the Type of a Naming Object ..................................... 19-13
  19.7.4  Get an Object’s Naming Version ..................................... 19-13
  19.7.5  Get an Object’s Subtype ID ............................................. 19-14
  19.7.6  Get an Object’s Checkout Status ...................................... 19-14
  19.7.7  Get Checkout Properties ............................................... 19-15
  19.7.8  Get a Naming Object’s Parent ......................................... 19-15
  19.7.9  Get a Parent Naming Object .......................................... 19-16
  19.7.10 Get the Latest Version of the Parent Object ...................... 19-16
  19.7.11 Get the Naming Status of a Parent Object...................... 19-17
  19.7.12 Get the Validation Status of a Parent Object .................... 19-18
  19.7.13 Get a Definition Object ................................................ 19-18
  19.7.14 Get a Lookup Meaning ................................................ 19-19
  19.7.15 Find Whether an Object is an Instance ......................... 19-19
  19.7.16 Find Whether Checked Out By Current User .................... 19-20
  19.7.17 Find Whether a Checkout is User-Specific ....................... 19-20
### 19.7.18 Find Whether Checkout is Implicit ................................................................. 19-21
### 19.8 Read Messages ..................................................................................................... 19-21
### 19.8.1 Get a Message .................................................................................................. 19-21
### 19.8.2 Get a Message Count ...................................................................................... 19-22
### 19.8.3 Initialize a Message Stack .............................................................................. 19-22

#### 20 Execution Setups

<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>20.1</td>
<td>Create and Modify Execution Setups</td>
<td>20-1</td>
</tr>
<tr>
<td>20.1.1</td>
<td>Create an Execution Setup</td>
<td>20-1</td>
</tr>
<tr>
<td>20.1.2</td>
<td>Check Out an Execution Setup</td>
<td>20-2</td>
</tr>
<tr>
<td>20.1.3</td>
<td>Modify an Execution Setup</td>
<td>20-3</td>
</tr>
<tr>
<td>20.1.4</td>
<td>Modify a Parameter</td>
<td>20-3</td>
</tr>
<tr>
<td>20.1.5</td>
<td>Modify an Execution Setup Parameter</td>
<td>20-4</td>
</tr>
<tr>
<td>20.1.6</td>
<td>Load Parameter Details</td>
<td>20-4</td>
</tr>
<tr>
<td>20.1.7</td>
<td>Copy an Execution Setup</td>
<td>20-5</td>
</tr>
<tr>
<td>20.1.8</td>
<td>Check In an Execution Setup</td>
<td>20-5</td>
</tr>
<tr>
<td>20.1.9</td>
<td>Submit an Execution Setup</td>
<td>20-6</td>
</tr>
<tr>
<td>20.1.10</td>
<td>Submit an Execution Setup for Instances</td>
<td>20-7</td>
</tr>
<tr>
<td>20.1.11</td>
<td>Submit an Execution Setup for Compound Objects</td>
<td>20-8</td>
</tr>
<tr>
<td>20.1.12</td>
<td>Upgrade an Execution Setup</td>
<td>20-9</td>
</tr>
<tr>
<td>20.1.13</td>
<td>Upgrade All Execution Setups</td>
<td>20-10</td>
</tr>
<tr>
<td>20.1.14</td>
<td>Make an Execution Setup Active</td>
<td>20-10</td>
</tr>
<tr>
<td>20.1.15</td>
<td>Remove an Execution Setup</td>
<td>20-11</td>
</tr>
</tbody>
</table>

#### 21 Mappings

<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>21.1</td>
<td>Create and Modify Mappings</td>
<td>21-1</td>
</tr>
<tr>
<td>21.1.1</td>
<td>Map a Column</td>
<td>21-1</td>
</tr>
<tr>
<td>21.1.2</td>
<td>Map a Table Descriptor to a Table Instance</td>
<td>21-2</td>
</tr>
<tr>
<td>21.1.3</td>
<td>Get a Table Instance ID</td>
<td>21-3</td>
</tr>
<tr>
<td>21.1.4</td>
<td>Create a Table Descriptor from a Table Instance</td>
<td>21-3</td>
</tr>
<tr>
<td>21.1.5</td>
<td>Create a Table Instance from a Table Descriptor</td>
<td>21-4</td>
</tr>
<tr>
<td>21.1.6</td>
<td>Modify a Mapping Column</td>
<td>21-4</td>
</tr>
<tr>
<td>21.1.7</td>
<td>Modify a Mapping at the Table Descriptor Level</td>
<td>21-5</td>
</tr>
<tr>
<td>21.1.8</td>
<td>Get a PRREF_ID for an Executable in a Workflow</td>
<td>21-5</td>
</tr>
<tr>
<td>21.1.9</td>
<td>Get a PRREF_ID for an Object in a Work Area</td>
<td>21-6</td>
</tr>
<tr>
<td>21.1.10</td>
<td>Get a PRREF_ID for a Program in a Report Set</td>
<td>21-7</td>
</tr>
</tbody>
</table>

#### 22 Outputs

<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>22.1</td>
<td>Generate Outputs</td>
<td>22-1</td>
</tr>
<tr>
<td>22.1.1</td>
<td>Submit a Print Request</td>
<td>22-1</td>
</tr>
<tr>
<td>22.1.2</td>
<td>Get an Output’s BLOB</td>
<td>22-2</td>
</tr>
<tr>
<td>22.1.3</td>
<td>Get an Output’s CLOB</td>
<td>22-2</td>
</tr>
</tbody>
</table>

#### 23 Version Labels

<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>23.1</td>
<td>Modify Version Labels</td>
<td>23-1</td>
</tr>
<tr>
<td>23.1.1</td>
<td>Update a Version Label</td>
<td>23-1</td>
</tr>
</tbody>
</table>
24 Classification

24.1 Classify Objects ........................................................................................................... 24-1
24.1.1 Classify an Object ................................................................................................... 24-1
24.1.2 Declassify an Object .............................................................................................. 24-2
24.2 Classify Subtypes ....................................................................................................... 24-2
24.2.1 Get a Subtype Classification Level ......................................................................... 24-3
24.2.2 Get an Object Classification Value ......................................................................... 24-3
24.3 Get a Parent Term ..................................................................................................... 24-4
24.3.1 Create and Modify Classification Hierarchy Values ................................................. 24-4
24.3.2 Insert a Classification Value ................................................................................... 24-5
24.3.3 Update a Classification Value ................................................................................ 24-5
24.3.4 Delete a Classification Value .................................................................................. 24-6

25 Job Execution

25.1 Create and Execute Output Jobs ................................................................................ 25-1
25.1.1 Create a Binary Output ........................................................................................ 25-1
25.1.2 Upload an Output BLOB ....................................................................................... 25-2
25.1.3 Upload an Output Clob .......................................................................................... 25-2
25.1.4 Upload a LOB to a Temporary Table .................................................................... 25-3
25.1.5 Download a Job Output BLOB ............................................................................. 25-4
25.1.6 Queue a Job ......................................................................................................... 25-4
25.1.7 Wait for a Job to Complete .................................................................................... 25-5
25.1.8 Generate an XML Payload .................................................................................... 25-5
25.2 Retrieve Information about Ongoing Jobs ................................................................ 25-7
25.2.1 Get an Ongoing Job ID ........................................................................................ 25-7
25.2.2 Get Currently Executing Parameters .................................................................... 25-7
25.2.3 Get Information About a Job ................................................................................ 25-8
25.2.4 Get Job Information (Overloaded) ...................................................................... 25-8
25.3 Set Execution Statuses ............................................................................................. 25-9
25.3.1 Set a User-specific Completion Status ................................................................. 25-9
25.3.2 Set a Customized Output Title ............................................................................. 25-10
25.3.3 Set a Customized Output Subtitle ....................................................................... 25-10
25.3.4 Set an Output Parameter ..................................................................................... 25-11
25.3.5 Get a Completion Status ...................................................................................... 25-11
25.4 Submit Messages ...................................................................................................... 25-11
25.4.1 Submit a Message ................................................................................................. 25-12
25.5 Create Submission Records ..................................................................................... 25-12
25.5.1 Start a Job ............................................................................................................. 25-12
25.5.2 Create a Submission ............................................................................................. 25-13
25.5.3 Create a Submission from a Job .......................................................................... 25-14
25.5.4 Add a Job Log ...................................................................................................... 25-14

26 Security Policy

26.1 Create and Modify Security Policies .......................................................................... 26-1
26.1.1 Create a Subtype .................................................................................................. 26-2
26.1.2 Copy a Subtype ................................................................................................. 26-2
26.1.3 Modify a Subtype .................................................................................................................. 26-3
26.1.4 Assign Roles to a Subtype Operation .................................................................................. 26-3
26.1.5 Assign Operations to a Subtype Role ................................................................................. 26-3
26.1.6 Remove a Subtype ................................................................................................................ 26-4
26.1.7 Create a Role ....................................................................................................................... 26-4
26.1.8 Modify a Role ....................................................................................................................... 26-5
26.1.9 Add a Group Role ................................................................................................................. 26-5
26.1.10 Get Roles for a User .......................................................................................................... 26-6
26.1.11 Remove a Role ................................................................................................................... 26-6
26.1.12 Remove a Group Role ....................................................................................................... 26-7
26.1.13 Create a User Group ......................................................................................................... 26-7
26.1.14 Add Users to a Group ....................................................................................................... 26-7
26.1.15 Remove Users from a Role in a User Group ..................................................................... 26-8
26.1.16 Assign a User Group to an Object .................................................................................... 26-8
26.1.17 Copy a User Group ........................................................................................................... 26-9
26.1.18 Copy a User Group with its Users ................................................................................... 26-9
26.1.19 Modify a User Group ....................................................................................................... 26-10
26.1.20 Remove All Group Roles from a User Group ................................................................. 26-10
26.1.21 Remove All Users in a Group .......................................................................................... 26-11
26.1.22 Revoke a User Group From an Object ............................................................................. 26-11
26.1.23 Undo a Revoke a User Group Action .............................................................................. 26-12
26.1.24 Remove a User Group ..................................................................................................... 26-12
26.1.25 Unassign a User Group From an Object ........................................................................ 26-13
26.1.26 Unassign Roles from an Operation on an Object’s Subtype .......................................... 26-13
26.1.27 Unassign Operations on an Object Subtype’s Role ....................................................... 26-14

27 Validation

27.1 Validate Objects .................................................................................................................... 27-1
27.1.1 Update an Object’s Validation Status .............................................................................. 27-1
27.2 Create and Modify Validation Supporting Documents ........................................................... 27-2
27.2.1 Create a Validation Supporting Document ...................................................................... 27-2
27.2.2 Update a Validation Supporting Document .................................................................. 27-2
27.2.3 Obsolete a Validation Supporting Document ............................................................... 27-3

Part IV Oracle Health Sciences Data Management Workbench APIs

28 Introduction to Oracle DMW APIs

28.1 Set Up Study Environment .................................................................................................. 28-1
28.1.1 Initialize a Study and Lifecycle ....................................................................................... 28-2
28.2 Create or Modify an Expression .......................................................................................... 28-2
28.3 Using APIs to Create Custom Programs ........................................................................... 28-3

29 Flags, Models and Actions

29.1 Flag-Related APIs ............................................................................................................... 29-1
29.1.1 Set Flag ......................................................................................................................... 29-1
29.1.2 Get Flag ....................................................................................................................... 29-2
30 Code Lists

30.1 Create and Modify Code Lists ................................................................. 30-1
30.1.1 Create a Code List ............................................................ 30-1
30.1.2 Modify a Code List ........................................................... 30-2
30.1.3 Remove a Code List .......................................................... 30-3
30.1.4 Check In a Code List ......................................................... 30-3
30.1.5 Check Out a Code List ..................................................... 30-4
30.1.6 Add Values to a Code List ............................................. 30-4
30.1.7 Remove Values from a Code List .................................. 30-5
30.1.8 Get Code List Details for a Given Column .................. 30-5

31 Validation Checks

31.1 Create and Modify Validation Checks and Batches ................................. 31-1
31.1.1 Create a Validation Check Batch ........................................ 31-1
31.1.2 Modify a Validation Check Batch ..................................... 31-2
31.1.3 Remove Validation Check Batch(es) ................................... 31-3
31.1.4 Create a Validation Check ................................................ 31-4
31.1.5 Update a Validation Check ................................................ 31-6
31.1.6 Install a Validation Check Batch ...................................... 31-8
31.1.7 Submit a Validation Check Batch ....................................... 31-9
31.1.8 Check In a Validation Check Batch ................................... 31-10
31.1.9 Check Out a Validation Check Batch ................................ 31-10
31.1.10 Undo Checkout For a Validation Check Batch .................. 31-11
31.1.11 Update Validation Status of a Validation Check Batch .......... 31-11
31.1.12 Upgrade a Validation Check Batch .................................. 31-12
31.1.13 Remove Validation Check(s) ........................................... 31-13
31.1.14 Enable or Disable Validation Checks ............................... 31-13
31.1.15 Reorder Validation Checks .............................................. 31-14
32 Transformations

32.1 Create and Modify Transformation Maps .............................................................. 32-1
32.1.1 Create Transformation Maps ........................................................................... 32-2
32.1.2 Modify Transformation Maps .......................................................................... 32-4
32.1.3 Mark Table Maps Not Used ............................................................................ 32-7
32.1.4 Mark Column Maps Not Used ......................................................................... 32-7
32.1.5 Check In Transformation Maps ....................................................................... 32-8
32.1.6 Check Out Transformation Maps .................................................................... 32-8
32.1.7 Undo Checkout Transformation Map ................................................................. 32-9
32.1.8 Auto Map Tables ............................................................................................. 32-9
32.1.9 Accept Table Mappings ................................................................................... 32-10
32.1.10 Auto Map Columns ....................................................................................... 32-11
32.1.11 Accept Column Mappings ............................................................................. 32-11
32.1.12 Upgrade Transformation Map ........................................................................ 32-12
32.1.13 Install Transformation Map ........................................................................... 32-12
32.1.14 Remove Transformation Map ......................................................................... 32-13
32.1.15 Validate Transformation Maps ....................................................................... 32-14
32.1.16 Update Validation Status ............................................................................... 32-14
32.1.17 Execute Transformation Map .......................................................................... 32-15
32.1.18 Create Staging Table ..................................................................................... 32-16
32.1.19 Validate Expression ...................................................................................... 32-18

Part V Oracle DMW Public Views

33 Public Views

33.1 Code Lists Views .................................................................................................. 33-1
33.1.1 DME_PUB_CODELIST_V ................................................................................. 33-1
33.1.2 DME_PUB_CODELIST_VALUES_V ................................................................. 33-1
33.2 Validation Checks Views ..................................................................................... 33-1
33.2.1 DME_PUB_VC_BATCHES_V ........................................................................... 33-1
33.2.2 DME_PUB_VCDETAILS_V ............................................................................. 33-2
33.3 Transformation Views .......................................................................................... 33-2
33.3.1 DME_PUB_DF_XFORM_MAP_V .................................................................... 33-2
33.3.2 DME_PUB_DF_MAP_ENTITY_V ...................................................................... 33-2
33.3.3 DME_PUB_XFM_SOURCE_TABLES_V .......................................................... 33-2
33.3.4 DME_PUB_XFM_SOURCE_COLUMNS_V ....................................................... 33-3
33.3.5 DME_PUB_XFM_TARGET_TABLES_V ............................................................ 33-3
33.3.6 DME_PUB_XFM_TARGET_COLUMNS_V ....................................................... 33-3
33.3.7 DME_PUB_XFM_AUTOMAPS_V ................................................................... 33-3
33.3.8 DME_PUB_XFM_COL_AUTOMAPS_V ............................................................ 33-3
33.3.9 DME_PUB_XFM_CUSTOM_PROGRAMS_V ....................................................... 33-3
33.3.10 DME_PUB_XFM_EXPR_STDFUNC_V ............................................................ 33-3
33.3.11 DME_PUB_XFM_EXPR_STATIC_PKGS_V ....................................................... 33-3
This book contains information on public Application Programming Interfaces (APIs) that can be used to interface with Oracle Life Sciences Data Hub and Oracle Health Sciences Data Management Workbench.

This preface contains the following topics:

- "Audience" on page xv
- "Documentation Accessibility" on page xv
- "Finding Information and Patches on My Oracle Support" on page xv
- "Finding Oracle Documentation" on page xvii
- "Related Documents" on page xviii
- "Conventions" on page xviii

Audience

This manual is intended for programmers with an understanding of Oracle technology.

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 customers have access to electronic support through My Oracle Support. For information, visit http://www.oracle.com/pls/topic/lookup?ctx=acc&id=info or visit http://www.oracle.com/pls/topic/lookup?ctx=acc&id=trs if you are hearing impaired.

Finding Information and Patches on My Oracle Support

Your source for the latest information about Oracle Clinical is Oracle Support's self-service Web site My Oracle Support (formerly MetaLink).

Before you install and use Oracle Clinical, always visit the My Oracle Support Web site for the latest information, including alerts, White Papers, installation verification (smoke) tests, bulletins, and patches.
Creating a My Oracle Support Account

You must register at My Oracle Support to obtain a user name and password account before you can enter the Web site.

To register for My Oracle Support:
1. Open a Web browser to https://support.oracle.com.
2. Click the Register link to create a My Oracle Support account. The registration page opens.
3. Follow the instructions on the registration page.

Signing In to My Oracle Support

To sign in to My Oracle Support:
1. Open a Web browser to https://support.oracle.com.
2. Click Sign In.
3. Enter your user name and password.
4. Click Go to open the My Oracle Support home page.

Finding Information on My Oracle Support

There are many ways to find information on My Oracle Support.

Searching by Article ID

The fastest way to search for information, including alerts, White Papers, installation verification (smoke) tests, and bulletins is by the article ID number, if you know it.

To search by article ID:
2. Locate the Search box in the upper right corner of the My Oracle Support page.
3. Click the sources icon to the left of the search box, and then select Article ID from the list.
4. Enter the article ID number in the text box.
5. Click the magnifying glass icon to the right of the search box (or press the Enter key) to execute your search.

The Knowledge page displays the results of your search. If the article is found, click the link to view the abstract, text, attachments, and related products.

Searching by Product and Topic

You can use the following My Oracle Support tools to browse and search the knowledge base:

- Product Focus — On the Knowledge page under Select Product, type part of the product name and the system immediately filters the product list by the letters you have typed. (You do not need to type "Oracle.") Select the product you want from the filtered list and then use other search or browse tools to find the information you need.
Advanced Search — You can specify one or more search criteria, such as source, exact phrase, and related product, to find information. This option is available from the Advanced link on almost all pages.

Finding Patches on My Oracle Support
Be sure to check My Oracle Support for the latest patches, if any, for your product. You can search for patches by patch ID or number, or by product or family.

To locate and download a patch:
2. Click the Patches & Updates tab. The Patches & Updates page opens and displays the Patch Search region. You have the following options:
   - In the Patch ID or Number field, enter the number of the patch you want. (This number is the same as the primary bug number fixed by the patch.) This option is useful if you already know the patch number.
   - To find a patch by product name, release, and platform, click the Product or Family link to enter one or more search criteria.
3. Click Search to execute your query. The Patch Search Results page opens.
4. Click the patch ID number. The system displays details about the patch. In addition, you can view the Read Me file before downloading the patch.
5. Click Download. Follow the instructions on the screen to download, save, and install the patch files.

Finding Oracle Documentation
The Oracle Web Site contains links to all Oracle user and reference documentation. You can view or download a single document or an entire product library.

Finding Oracle Health Sciences Documentation
For the latest user documentation on Oracle Life Sciences Data Hub, go to the Oracle Health Sciences—Clinical documentation page at:

Note: Always check the Oracle Health Sciences Documentation page to ensure you have the latest updates to the documentation.

Finding Other Oracle Documentation
To get user documentation for other Oracle products:
1. Go to the following Web page:
   http://www.oracle.com/technology/documentation/index.html
   Alternatively, you can go to http://www.oracle.com, point to the Support tab, and then click Documentation.
2. Scroll to the product you need and click the link.
3. Click the link for the documentation you need.
Related Documents

This section lists the documents in the documentation set, followed by their part number. The most recent version of each guide is posted on the Oracle website; see "Finding Oracle Documentation" on page xvii.

LSH Documentation Set Available on Oracle Website

■ Oracle Life Sciences Data Hub Installation Guide (Part E35295)
■ Oracle Life Sciences Data Hub Implementation Guide (Part E35296)
■ Oracle Life Sciences Data Hub System Administrator’s Guide (Part E35297)
■ Oracle Life Sciences Data Hub Application Developer’s Guide (Part E35298)
■ Oracle Life Sciences Data Hub User’s Guide (Part E35305)
■ Oracle Life Sciences Data Hub Application Programming Interface Guide (Part E35306)
■ Oracle Life Sciences Data Hub Adapter Toolkit Guide (Part E35307)

DMW Documentation Set Available on Oracle Website

■ Oracle Health Sciences Data Management Workbench Installation Guide (Part E35223)
■ Oracle Health Sciences Data Management Workbench User’s Guide (Part E35217)
■ Oracle Health Sciences Data Management Workbench and Life Sciences Data Hub Security Guide (Part E38924)

Conventions

The following text conventions are used in this document:

<table>
<thead>
<tr>
<th>Convention</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>boldface</strong></td>
<td>Boldface type indicates graphical user interface elements associated with an action, or terms defined in text or the glossary.</td>
</tr>
<tr>
<td>Initial Capitalization for defined objects</td>
<td>User-defined objects in Oracle LSH such as Tables, Source Code, and Variables, have initial capitalization to distinguish them from generic tables, source code, and variables, for example.</td>
</tr>
<tr>
<td><em>italic</em></td>
<td>Italic type indicates book titles, emphasis, or placeholder variables for which you supply particular values.</td>
</tr>
<tr>
<td><strong>monospace</strong></td>
<td>Monospace type indicates commands within a paragraph, URLs, code in examples, text that appears on the screen, or text that you enter.</td>
</tr>
</tbody>
</table>
Part I

Essential Information

This part of the Oracle Life Sciences Data Hub (Oracle LSH) API guide contains the basic information you need on running APIs.

Part I contains the following chapters:

- Chapter 1, "Using Application Programming Interfaces"
- Chapter 2, "Reference Information"
This section contains the following topics:

- **Section 1.1, "About APIs"**
- **Section 1.2, "Calling APIs from Outside the Oracle Life Sciences Data Hub"**
- **Section 1.3, "Calling APIs from Defined Programs"**
- **Section 1.4, "Code Example Using Security and Error Message APIs"**

**Views**  
All Oracle Life Sciences Data Hub (Oracle LSH) views are public and have names that begin with "cdr_". You can see them in SQL Developer or a similar tool, or query for them using the string `cdr_`

---

**Note:** During its initial development, Oracle LSH was known as CDR. Therefore many internal names contain the string `cdr`. Please think of CDR as a synonym for LSH.

### 1.1 About APIs

Oracle LSH includes a set of APIs that enable you to do most of the things you can do through the user interface, including creating, modifying, and installing objects.

You can call Oracle LSH APIs from source code in a defined Program in Oracle LSH. In this case, no additional security or setup is required.

If you have an Oracle LSH database account with certain privileges, you can also develop programs that call APIs in a tool outside of Oracle LSH; such as SAS, Oracle SQL Developer, or SQL*Plus. You can then see views of all the Oracle LSH data you need, including data from both the LSH (CDR) schema and, for classification data, the TMS schema. You can make the programs you write available to other people from the external tool. See "Calling APIs from Outside the Oracle Life Sciences Data Hub" on page 1-3.

**Example 1: Using APIs to Perform Multiple Tasks at Once**  
You can write a package that calls multiple APIs to do with one execution what it would take many tasks in the user interface (UI) to do; for example, create a Domain, an Application Area inside the Domain, a Work Area inside the Application Area, and multiple Load Sets, Tables, and Programs, each with a definition in the Application Area and an instance in the Work Area, and install the Work Area. If you have a standard structure for Project/Therapeutic Area Domains, for example, you may want to work this way. However, remember that you can also copy a Domain and all its contents at once in the user interface.
Using APIs is even more attractive when you want to create, for example, multiple objects with variations or large complex objects such as Report Sets. You can create a spreadsheet to store all the variable information and load its data into an Oracle LSH Table instance using a Text Load Set. In your program, use a loop to read all the spreadsheet data and call the relevant Oracle LSH APIs to create the objects.

Example 2: Calling APIs from an External System’s UI  You may want to allow people in your company to perform actions on Oracle LSH objects from an external system.

For example, instead of requiring that SAS developers check out Source Code in Oracle LSH before opening the SAS IDE from an Oracle LSH Program, you may want to add a button to the SAS user interface that calls the API for checking out the Source Code object when clicked. Then, if the program is located in a schema with Execute privileges on the security API, any user with SAS, a database account in Oracle LSH, and normal Oracle LSH object security privileges on the Source Code definition, can check out the Source Code definition directly from SAS.

Understand Oracle LSH Functionality  To use Oracle LSH APIs, you must understand basic Oracle LSH functionality including:

- **Object Ownership.** You must create container objects before creating the objects they contain, because to create any object you must identify its namespace (parent, or container) object. For example, begin by defining a Domain, then an Application Area, then a Work Area, and then create a Table definition in the Application Area and an instance of it in the Work Area. You can use a single API to create both the Table definition and an instance of it. For details, see "Object Ownership" in the Oracle Life Sciences Data Hub Application Developer’s Guide.

- **Installation.** You must create an instance of an object definition and install it before you can execute or otherwise use the object.

- **Mapping.** All executable objects must contain at least one Table Descriptor, each of which must be mapped to an installed Table instance. For details, see "Defining and Mapping Table Descriptors" in the Oracle Life Sciences Data Hub Application Developer’s Guide.

- **Checking Objects In and Out.** You must check objects out to modify them and check them in before you install and use them. For details, see “Understanding Object Versions and Checkin/Checkout” in the Oracle Life Sciences Data Hub Application Developer’s Guide.

- **Security.** All objects require user group assignments to control user access. For details, see “Applying Security to Objects and Outputs” in the Oracle Life Sciences Data Hub Application Developer’s Guide.

- **Classification.** To enable objects to appear in the Reports tab of the user interface for end users to run them and view their outputs, you must classify them. Classifications can also be used in searching for objects. For details, see “Classifying Objects and Outputs” in the Oracle Life Sciences Data Hub Application Developer’s Guide.

- **Validation.** Objects should be validated according to your company policy whether they are created in the user interface or with APIs. For details, see “Validating Objects and Outputs” in the Oracle Life Sciences Data Hub Application Developer’s Guide.

- **Object-Specific Information.** Further information on each object type is included in other chapters of the Oracle Life Sciences Data Hub Application Developer’s Guide.
1.2 Calling APIs from Outside the Oracle Life Sciences Data Hub

This section contains the following topics:

- Section 1.2.1, "Security Setup Required"
- Section 1.2.2, "Calling the Security API Package"
- Section 1.2.3, "Calling APIs from SAS"
- Section 1.2.4, "Using a Permanent Schema for Deploying Programs that Call APIs"

1.2.1 Security Setup Required

To run any API package from a tool outside of Oracle LSH, such as SAS, SQL Developer, or SQL*Plus, your system administrator needs to do the following:

1. Set up an Oracle LSH database account linked to your LSH user account; see "Creating Database Accounts" in the Oracle Life Sciences Data Hub System Administrator’s Guide.

2. Grant your Oracle LSH database account Execute privileges on the API security package cdr_pub_api_initialization.

In addition, to run APIs that insert, delete, or modify classification hierarchy terms, you need security access to the Oracle Thesaurus Management System (TMS) instance that is installed as part of Oracle LSH. The Oracle LSH classification system is based on TMS. Ask your system administrator to use the script tmsadduser.sql to do the following:

3. Create a TMS user account with the same name as your Oracle LSH database account so that your account is entered in the TMS_ACCOUNTS and OPA_ACCOUNTS tables.

4. Give your TMS user account superuser privileges in the TMS ACCOUNTS table.

5. Grant your TMS user account the TMS_MAINTAIN_PRIV database role.

1.2.2 Calling the Security API Package

For every program that you run from outside Oracle LSH to call an Oracle LSH API, you must first call a special security API: cdr_pub_api_initialization. This API contains three functions:

- EnableApis
- DisableApis
- AreApisEnabled

When you initialize any API, the AreApisEnabled function of the security API, cdr_pub_api_initialization, is called to check the calling program. If the program does not have the EnableAPIs flag set to True, the initialization fails.

To set the EnableApis flag to True, call the EnableApis function of the same security API from your program. To call the EnableApis function, you need a schema/user account with an Execute privilege on the cdr_pub_api_initialization API granted by a system administrator.

Therefore, when you write a program that calls an API and is intended for use outside Oracle LSH, set the EnableApis flag to True in your program and then set it to False at the end to force the security check on the schema the next time the program is run:

1. Begin the body with the following code to call the function to enable APIs:
call cdr_pub_api_initialization.enableApis (arguments);

The arguments are described in cdr_pub_api_initialization itself.

2. At the end of the body, disable APIs with the following code:

cdr_pub_api_initialization.disableApis (arguments);

See Example 1–1, "Program that Calls the API to Define a Work Area and Calls the Security and Error Message APIs" on page 1-5.

1.2.3 Calling APIs from SAS

If you need to call multiple APIs from SAS, you may want to use a PL/SQL wrapper around the API calls so that you only call PL/SQL once. The Oracle Life Sciences Data Hub Application Developer’s Guide has two examples. In the Report Sets chapter, see "Passing Values from a Program Instance to the Report Set for Post-Processing" and in the Programs chapter see "Calling an API to Capture Output Parameter Values."

1.2.4 Using a Permanent Schema for Deploying Programs that Call APIs

When you develop a program outside Oracle LSH that will call Oracle LSH APIs, you can use your own schema in the external tool (such as SQL*Plus, SQL Developer, or SAS) to run and test the program, if you have Execute privileges on cdr_pub_api_initialization. When you are ready to allow other people to run it, copy it into a different location.

Oracle recommends setting up one or more permanent, publicly available schemas in the Oracle LSH database for the purpose of compiling and storing programs that call Oracle LSH APIs. Grant each schema Execute privileges on cdr_pub_api_initialization. This approach has the following advantages:

- If a user manually runs your program, he or she must enter the program location and name explicitly. This will be much easier if the user knows which schema contains such programs.
- If you set up the program to run automatically when a user clicks a button in the external system’s user interface, for example, you must hardcode the program’s name and location into the code.
- You can grant Execute on cdr_pub_api_initialization to a controlled number of schemas.

1.3 Calling APIs from Defined Programs

If you develop and run a Program that calls an API within Oracle LSH—that is, in the defined Source Code of a defined Program object—no security is required beyond normal Oracle LSH object security. You do not need Execute privileges on the cdr_pub_api_initialization API, and you do not need to enable APIs in your Program code.

Note: Within Oracle LSH, the calls to cdr_pub_api_initialization are unnecessary and in fact a program that includes such a call will not compile because the Work Area schema does not have Execute privileges on cdr_pub_api_initialization.

You do need to install the Program before you can run it, as you do any defined Program in Oracle LSH.
You can write packages in an Oracle LSH Program that do anything with APIs that you could do in a package outside Oracle LSH. For example, you could create an instance of a Program definition whose Source Code created a Work Area, several Load Sets, and a Program to merge the data, instead of defining the Work Area, Load Sets and Program through the Oracle LSH user interface.

### 1.4 Code Example Using Security and Error Message APIs

There are two utility Oracle LSH APIs that you call in conjunction with other Oracle LSH APIs:

- **cdr_pub_api_initialization**. This API is required for developing and running programs that call any Oracle LSH API from outside Oracle LSH. See Section 1.2.2, "Calling the Security API Package" for further information.

- **cdr_pub_msg_pub**. This API returns error messages from other Oracle LSH APIs called in the same package.

The following code provides an example of calling the API to define a Work Area and each of the utility APIs.

**Example 1–1  Program that Calls the API to Define a Work Area and Calls the Security and Error Message APIs**

```sql
CDR_PUB_DF_WORKAREA.CREATEWORKAREA (  
P_API_VERSION=>1,  
P_INIT_MSG_LIST=>CDR_PUB_DEF_CONSTANTS.G_FALSE,  
P_COMMIT=>CDR_PUB_DEF_CONSTANTS.G_FALSE,  
P_VALIDATION_LEVEL=>CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,  
X_RETURN_STATUS => X_RETURN_STATUS,  
X_MSG_COUNT    => X_MSG_COUNT,  
X_MSG_DATA     => X_MSG_DATA,  
PIO_SOURCECDRNAMING =>VARWANSOBJ,  
PIO_WORKAREAOBJTYPE =>VARWAOBJ,  
PI_DEFCLASSIFICATIONCOLL => NULL);

IF X_RETURN_STATUS <> 'S' THEN  
    DBMS_OUTPUT.PUT_LINE('ERROR FOUND IN CREATEPROGRAM');  
END IF ;

X_MSG_COUNT := CDR_PUB_MSG_PUB.COUNT_MSG(  
P_API_VERSION=>1,  
P_INIT_MSG_LIST=>CDR_PUB_DEF_CONSTANTS.G_FALSE,  
P_COMMIT=>CDR_PUB_DEF_CONSTANTS.G_FALSE,  
P_VALIDATION_LEVEL=>CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL);

IF X_MSG_COUNT >= 1 THEN  
    FOR I IN 1..X_MSG_COUNT LOOP  
        IF I =1 THEN  
            X_MSG_DATA := CDR_PUB_MSG_PUB.GET(  
                P_API_VERSION=>1,  
P_INIT_MSG_LIST=>CDR_PUB_DEF_CONSTANTS.G_FALSE,  
P_COMMIT=>CDR_PUB_DEF_CONSTANTS.G_FALSE,  
P_VALIDATION_LEVEL=>CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,  
P_MSG_INDEX =>CDR_PUB_MSG_PUB.G_FIRST,  
P_ENCODED =>CDR_PUB_DEF_CONSTANTS.G_FALSE);  
            ELSIF I = X_MSG_COUNT THEN  
                X_MSG_DATA := CDR_PUB_MSG_PUB.GET(  
                    P_API_VERSION=>1,  
P_INIT_MSG_LIST=>CDR_PUB_DEF_CONSTANTS.G_FALSE,  
P_COMMIT=>CDR_PUB_DEF_CONSTANTS.G_FALSE,  
P_VALIDATION_LEVEL=>CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,  
P_MSG_INDEX =>CDR_PUB_MSG_PUB.G_FIRST,  
P_ENCODED =>CDR_PUB_DEF_CONSTANTS.G_FALSE);

        ELSE  
            X_MSG_DATA := CDR_PUB_MSG_PUB.GET(  
                P_API_VERSION=>1,  
P_INIT_MSG_LIST=>CDR_PUB_DEF_CONSTANTS.G_FALSE,  
P_COMMIT=>CDR_PUB_DEF_CONSTANTS.G_FALSE,  
P_VALIDATION_LEVEL=>CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,  
P_MSG_INDEX =>CDR_PUB_MSG_PUB.G_FIRST,  
P_ENCODED =>CDR_PUB_DEF_CONSTANTS.G_FALSE);

    END IF ;

    DBMS_OUTPUT.PUT_LINE('ERROR MESSAGE: ' || X_MSG_DATA);

    IF X_MSG_COUNT > 1 THEN  
        FOR I IN (2..X_MSG_COUNT-1) LOOP  
            X_MSG_DATA := CDR_PUB_MSG_PUB.GET(  
                P_API_VERSION=>1,  
P_INIT_MSG_LIST=>CDR_PUB_DEF_CONSTANTS.G_FALSE,  
P_COMMIT=>CDR_PUB_DEF_CONSTANTS.G_FALSE,  
P_VALIDATION_LEVEL=>CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,  
P_MSG_INDEX =>CDR_PUB_MSG_PUB.G_FIRST,  
P_ENCODED =>CDR_PUB_DEF_CONSTANTS.G_FALSE);

            DBMS_OUTPUT.PUT_LINE('ERROR MESSAGE: ' || X_MSG_DATA);

        END LOOP ;

    END IF ;

END LOOP ;
```

Using Application Programming Interfaces  1-5
P_MSG_INDEX => CDR_PUB_MSG_PUB.G_LAST,
P_ENCODED => CDR_PUB_DEF_CONSTANTS.G_FALSE);
ELSE
  X_MSG_DATA := CDR_PUB_MSG_PUB.GET(
    P_API_VERSION=>1,
    P_INIT_MSG_LIST=>CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_COMMIT=>CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_VALIDATION_LEVEL=>CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
    P_MSG_INDEX => CDR_PUB_MSG_PUB.G_NEXT,
    P_ENCODED => CDR_PUB_DEF_CONSTANTS.G_FALSE);
END IF;
DBMS_OUTPUT.PUT_LINE('MESSAGE:' || I || ' : ' || X_MSG_DATA);
END LOOP;
END IF;
CDR_PUB_API_INITIALIZATION.DISABLEAPIS(
  P_API_VERSION=>1,
  P_INIT_MSG_LIST=>CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_COMMIT=>CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_VALIDATION_LEVEL=>CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
  X_RETURN_STATUS  => X_RETURN_STATUS,
  X_MSG_COUNT    => X_MSG_COUNT,
  X_MSG_DATA    => X_MSG_DATA);
EXCEPTION
  WHEN OTHERS THEN
    CDR_PUB_API_INITIALIZATION.DISABLEAPIS(
      P_API_VERSION=>1,
      P_INIT_MSG_LIST=>CDR_PUB_DEF_CONSTANTS.G_FALSE,
      P_COMMIT=>CDR_PUB_DEF_CONSTANTS.G_FALSE,
      P_VALIDATION_LEVEL=>CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
      X_RETURN_STATUS  => X_RETURN_STATUS,
      X_MSG_COUNT    => X_MSG_COUNT,
      X_MSG_DATA    => X_MSG_DATA);
END MY_PROCEDURE;
BEGIN -- PACKAGE INIT BLOCK
  CDR_PUB_API_INITIALIZATION.ENABLEAPIS(
    P_API_VERSION=>1,
    P_INIT_MSG_LIST=>CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_COMMIT=>CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_VALIDATION_LEVEL=>CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
    X_RETURN_STATUS  => X_RETURN_STATUS,
    X_MSG_COUNT    => X_MSG_COUNT,
    X_MSG_DATA    => X_MSG_DATA);
END MY_PACKAGE;
This section contains the following topics:

- Section 2.1, "CDR Naming Version Object Type"
- Section 2.2, "CDR Base Object Type"
- Section 2.3, "CDR Object-Specific Database Object Types"
- Section 2.4, "Retrieving Reference Codelist Names and Values"
- Section 2.5, "Retrieving the Instance Domain ID"
- Section 2.6, "Standard Parameters"

### 2.1 CDR Naming Version Object Type

Object information is stored in two tables in the Oracle LSH database: `cdr_namings`, which contains one row for each defined Oracle LSH object, and `cdr_naming_versions`, which contains one row for each version of each defined Oracle LSH object. Information from these two tables is stored in two composite database object types: `cdr_naming_version_obj_type` and `cdr_base_obj_type`.

For both the composite object types, the attributes `company_id`, `obj_id`, `obj_ver`, `namespace_obj_id`, and `namespace_obj_ver` form a composite primary key. You can refer to any existing object using this primary key.

Parameters of type `cdr_naming_version_obj_type` are required in APIs for creating and modifying an object.

The attributes of `cdr_naming_version_obj_type` are:

- **company_id**: To get your company ID, use `CDR_PUB_DEF_FACTORY_UTILS.GetCompanyId`.
- **obj_id**: The unique ID of the object. Oracle LSH generates this ID when you create a new object.
- **obj_ver**: The object’s version number.

**Note**: The attributes `company_id`, `obj_id`, `obj_ver`, `namespace_obj_id`, and `namespace_obj_ver` together constitute an object’s primary key.

- **namespace_obj_id**: The unique ID of the object’s parent object; for example, a Table instance is always contained in a Work Area, so its `namespace_obj_id` is the object ID of its Work Area.
- **namespace_obj_ver**: The version number of the object’s parent object.
Note: You can create a child object only in the latest version of its parent object. If you pass a namespace version number that is not the latest when creating a child object, the system ignores the value you pass and creates the child in the latest version of the parent.

- **namespace_start_obj_ver**. This attribute contains the version number of the parent object at the time the version represented by `obj_ver` of the object represented by `obj_id` was created.

- **namespace_end_obj_ver**. This attribute contains the version number of the parent object at the time when the version represented by `obj_ver` of the object represented by `obj_id` was superseded by a higher version. If the object is still the most current version, then this attribute contains the value 999999.

- **object_type_rc**. This attribute defines what type of object you are creating or modifying. This value is mandatory for creating objects, but not for modifying objects. See "Retrieving Reference Codelist Names and Values" on page 2-4 for information on retrieving valid values.

- **name**. This is the name of the object.

- **owning_location_rc**. This attribute is entered in the system at LSH installation time and is stored as a profile in the system. The system automatically sets this value to the profile value for all objects.

- **checked_out_flag_rc**. This value indicates whether the object is currently checked out or not. The possible values are $YESNO$YES and $YESNO$NO.

- **checked_out_id** is the user ID of the person who checked out the object, if it is currently checked out.

- **object_subtype_id**. This attribute specifies the ID of the object’s subtype. Use `CDR_PUB_DF_NAMING_UTIL.GetObjectSubtypeID` to retrieve an object’s subtype ID.

- **description**. This is an optional attribute but it is highly recommended that you provide a description for future reference. You can modify the description using appropriate API for the object.

- **ref_company_id**. If the object is an instance object, this attribute contains the company ID of the source definition.

- **ref_obj_id**. If the object is an instance object, this attribute contains the object ID of the source definition.

- **ref_obj_ver**. If the object is an instance object, this attribute contains the object version number of the source definition.

- **copied_from_company_id**. If the object is a copy of another object, this attribute contains the company ID of the original object.

- **copied_from_obj_id**. If the object is a copy of another object, this attribute contains the object ID of the original object.

- **copied_from_obj_ver**. If the object is a copy of another object, this attribute contains the object version number of the original object.

- **object_version_number**. This attribute is for Oracle LSH internal use only. Never enter a value for this attribute.
CDR Object-Specific Database Object Types

- **status_rc**. This attribute contains the current status of the object. See "Retrieving Reference Codelist Names and Values" on page 2-4 for information on retrieving valid values.
- **validation_status_rc**. This attribute contains the current validation status of the object. See "Retrieving Reference Codelist Names and Values" on page 2-4 for information on retrieving valid values.
- **version_label**. This attribute stores the version label of the object, if any.

### 2.2 CDR Base Object Type

For some operations on objects, only the identification contained in a CDR base object type (cdr_base_obj_type) is required. Some APIs allow you to operate on multiple objects at the same time by using a parameter based on a collection of CDR base object types called cdr_base_obj_coll.

A CDR Base Object Type contains a subset of the information contained in a CDR naming Version Object Type (see "CDR Naming Version Object Type" on page 2-1).

### 2.3 CDR Object-Specific Database Object Types

Each Oracle LSH object type has its own unique attributes beyond what is included in the CDR Naming Version Object Type and CDR Base Object Type. These unique attributes are included in a view for each object type. The view includes information on both definitions and instances of a particular object type. In the case of Tables, it includes Table Descriptors as well as Table definitions and instances.

APIs that are used to create or modify Oracle LSH defined objects contain parameters based on these supplementary database object types. You can set values for the object-specific attributes using these parameters.

For example, the supplementary database object type for Oracle LSH Programs is called cdr_program_obj_type. In the Create Program API, the parameter pi_cdrprgobjtype is of this type. Its attributes are:

- **company_id**. To get your company ID, use CDR_PUB_DEF_FACTORY_UTILS.GetCompanyId.
- **obj_id**. The unique ID of the Program.
- **obj_ver**. The Program's version number.
- **tech_type_id**. Different executable object types have different technology types, which can be queried using the view cdr_tech_types_v. Use the column program_type_rc to see which tech type is valid for a particular object type. In the case of Programs, only the tech types whose value in the program_type_rc column is $PROGRAMTYPES$PROGRAM and which are present in the lookup type cdr_tech_types are allowed. They are: $TECHTYPES$SAS, $TECHTYPES$SASCATALOGS, $TECHTYPES$SASFORMATS, $TECHTYPES$PLSQL, $TECHTYPES$REPORTS.

**Note:** Tech types that are not included in the lookup type cdr_tech_types are used internally only and should not be used with public APIs.

- **manual_validation_flag_rc**. This flag determines whether a Program's outputs receive their validation status from their Execution Setup or must be validated manually. The valid values are: $YESNO$YES and $YESNO$NO.

Reference Information 2-3
See the chapter on "Defining Programs" in the Oracle Life Sciences Data Hub Application Developer’s Guide for information about these attributes. Each object type has its own chapter in this manual where its attributes are described.

2.4 Retrieving Reference Codelist Names and Values

Some database object type attributes (those ending in the string _rc) have a fixed set of allowed values stored in a lookup (reference codelist). These attributes correspond to fields in the user interface with a drop-down or pop-up list of values. To supply or change one of these values you must enter the exact string stored in the reference codelist, with the codelist name surrounded by dollar signs and followed by a codelist value.

For example, the API to create any object includes a parameter of type cdr_naming_version_obj_type, one of whose attributes is object_type_rc. You must enter the correct string for the type of object you want to create.

Reference codelists are stored in a table you access through the view cdr_lookups. The following columns contain the following information:

- **lookup_type**: reference codelist names
- **lookup_code**: reference codelist values
- **meaning**: the text that is displayed in the user interface
- **description**: additional information (sometimes)

If you have LSH Setup Admin privileges you can look up reference codelists in the Applications user interface; see "Querying and Viewing Lookups" in the Oracle Life Sciences Data Hub System Administrator’s Guide.

You can browse the view in a tool like SQL Developer to find these values. However, it is not always easy to guess the name of the reference codelist. In that case, you can go into the Oracle LSH user interface to where they are displayed and note one of the allowed values, then query.

For example, object types are displayed in the Add drop-down list in the Work Area Properties screen. You can see that one object type is Business Area, so you can use the following query:

```sql
SELECT lookup_type, lookup_code, meaning FROM cdr_lookups WHERE meaning like '%Business Area%';
```

Now you know that the lookup_type for object types is CDR_OBJECT_TYPES and you can use the following query to get all the other values:

```sql
SELECT distinct lookup_code, meaning FROM cdr_lookups where lookup_type = 'CDR_OBJECT_TYPES';
```

2.5 Retrieving the Instance Domain ID

While working with classification-related APIs, you may need the domain ID for the Oracle LSH environment you are working in. This is the ID for your Oracle LSH instance, which is created during installation. It has nothing to do with user-defined Domains that contain Application Areas.

Use the following query to find the domain ID of your Oracle LSH environment:

```sql
SELECT def_domain_id FROM tms.tms_def_domains WHERE name = 'cdr_user_hier';
```
2.6 Standard Parameters

Some or all of the following standard Oracle Applications parameters are included in each function and procedure:

**P_API_VERSION** (Mandatory) Enter the current version of the API you are calling. The version number is displayed at the top of this page. The API compares the version numbers of incoming calls to its current version number and returns an error if they are incompatible.

**P_INIT_MSG_LIST** (Optional) Accept the default value (FND_API.G_FALSE) to ensure that this individual API does not initialize the message list when the procedure is entered. Pass FND_API.G_TRUE to override the default behavior.

**P_COMMIT** (Optional) Accept the default value (FND_API.G_FALSE) to ensure that this individual API does not commit upon completion. Pass FND_API.G_TRUE to override the default behavior.

**P_VALIDATION_LEVEL** (Optional) Accept the default value to perform full validation. No other values are currently supported.

**X_RETURN_STATUS** This output parameter returns the end status of the API: (S) Success, (E) Error or (U) Unexpected Error.

**X_MSG_COUNT** This output parameter returns the count of error messages if the return status is other than Success.

**X_MSG_DATA** This output parameter returns the text of the error message, if the message count is 1. If there are more than one messages, use cdr_pub_msg_pub.get to retrieve the messages.
This part of the Oracle Life Sciences Data Hub (Oracle LSH) API guide contains APIs that you can use to create, modify, check in or out, delete, copy, and move defined Oracle LSH objects. You can also perform object-specific tasks using these APIs.

Part II contains the following chapters:

- Chapter 3, "Application Areas"
- Chapter 4, "Business Areas"
- Chapter 5, "Data Marts"
- Chapter 6, "Domains"
- Chapter 19, "Setup Utilities"
- Chapter 7, "Load Sets"
- Chapter 8, "Parameter Sets"
- Chapter 9, "Planned Outputs"
- Chapter 10, "Programs"
- Chapter 11, "Report Sets"
- Chapter 12, "Software Source Codes"
- Chapter 13, "Tables"
- Chapter 14, "Parameters"
- Chapter 15, "Variables"
- Chapter 16, "Work Areas"
- Chapter 17, "Workflows"
- Chapter 18, "Workflow Notifications"
Application Areas

This is a public interface for Application Area-related operations—creating, modifying, and removing Application Areas; as well as copying and moving object definitions into an Application Area.

3.1 Define and Modify Application Areas

This section contains the following topics:

- Section 3.1.1, "Create an Application Area"
- Section 3.1.2, "Modify an Application Area"
- Section 3.1.3, "Remove an Application Area"
- Section 3.1.4, "Copy Objects into an Application Area"
- Section 3.1.5, "Move Objects into an Application Area"

3.1.1 Create an Application Area

Use this API to create a new Application Area.

**Name**  CDR_PUB_DF_APPLICATIONAREA.CreateApplicationArea

**Signature:**

PROCEDURE CREATEAPPLICATIONAREA(
    P_API_VERSION IN NUMBER,
    P_INIT_MSG_LIST IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_COMMIT IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_VALIDATION_LEVEL IN NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
    X_RETURN_STATUS OUT VARCHAR2,
    X_MSG_COUNT OUT NUMBER,
    X_MSG_DATA OUT VARCHAR2,
    PIO_SOURCECDRNAMING IN OUT CDR_NAMING_VERSION_OBJ_TYPE,
    PI_DEFCLASSIFICATIONCOLL IN CDR_CLASSIFICATIONS_COLL
);

**Parameters:** This API has standard (see "Standard Parameters" on page 5) as well as the following parameters:

- **PIO_SOURCECDRNAMING** (Mandatory) This is a parameter of table type CDR_NAMING_VERSION_OBJ_TYPE that contains object attributes. Enter values for the Application Area you want to create.
The following attributes are required: company_id, namespace_obj_id, namespace_obj_ver, object_type_rc. for object_type_rc enter $objcotypes$applarea.

- **PI_DEFCLASSIFICATIONCOLL** (Optional) By default the new Application Area is classified according to the subtype you assigned it in the CDR_NAMING_VERSION_OBJ_TYPE. If you want to override the default classifications for one or more classification levels, use this parameter. This is a collection of CDR_CLA_OBJ_TYPES, which have 5 attributes, including CLA_LEVEL_ID and CLASSIFICATION_ID. If you want the Application Area to inherit its classifications for a particular level from its parent Domain, enter the classification level ID and, for the CLASSIFICATION_ID, enter 0 (zero). If you want to explicitly assign one or more terms for a particular level, initialize a CDR_CLA_OBJ_TYPE for each term, entering the classification level ID and, for the CLASSIFICATION_ID, the term ID. The PAR_attributes are not relevant to Application Areas. Do not enter any values for them.

### 3.1.2 Modify an Application Area

Use this API to modify the name or description of an existing Application Area.

**Name**  
CDR_PUB_DF_APPLICATIONAREA.ModifyApplicationArea

**Signature**

```sql
PROCEDURE MODIFYAPPLICATIONAREA(
    P_API_VERSION IN NUMBER,
    P_INIT_MSG_LIST IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_COMMIT IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_VALIDATION_LEVEL IN NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
    X_RETURN_STATUS OUT VARCHAR2,
    X_MSG_COUNT OUT NUMBER,
    X_MSG_DATA OUT VARCHAR2,
    PIO_SOURCECDRNAMING IN OUT CDR_NAMING_VERSION_OBJ_TYPE
);
```

**Parameters**  
This API has standard (see "Standard Parameters" on page 5) as well as the following:

- **PIO_SOURCECDRNAMING** (Mandatory) This is a parameter of table type CDR_NAMING_VERSION_OBJ_TYPE that contains object attributes. Enter values to identify the Application Area you want to modify and for the attributes you want to modify. You can modify the name and description. All attributes are required.

### 3.1.3 Remove an Application Area

Use this API to remove one or more Application Areas and all its/their contents.

**Name**  
CDR_PUB_DF_APPLICATIONAREA.RemoveApplicationAreas

**Signature**

```sql
PROCEDURE REMOVEAPPLICATIONAREAS(
    P_API_VERSION IN NUMBER,
    P_INIT_MSG_LIST IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_COMMIT IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_VALIDATION_LEVEL IN NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
    X_RETURN_STATUS OUT VARCHAR2,
    X_MSG_COUNT OUT NUMBER,
    X_MSG_DATA OUT VARCHAR2,
);
```
### Parameters

This API has standard (see "Standard Parameters" on page 5) as well as the following parameters:

- **PI_BASEOBJCOLL** (Mandatory) This is a collection of CDR_BASE_OBJ_TYPES. For each Application Area that you want to remove (including all its contents), initialize a CDR_BASE_OBJ_TYPE and then extend the collection.

The following attributes are required: COMPANY_ID, OBJ_ID, OBJ_VER, OBJECT_VERSION_NUMBER, NAMESPACE_OBJ_ID, NAMESPACE_OBJ_VER.

### 3.1.4 Copy Objects into an Application Area

Use this API to copy object definitions into an Application Area.

**Name**  
CDR_PUB_DF_APPLICATIONAREA.CopyObjectIntoAA

**Signature**

```
PROCEDURE COPYOBJECTINTOAA(
  P_API_VERSION  IN    NUMBER,
  P_INIT_MSG_LIST  IN    VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_COMMIT  IN    VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_VALIDATION_LEVEL  IN    NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
  X_RETURN_STATUS  OUT    VARCHAR2,
  X_MSG_COUNT  OUT    NUMBER,
  X_MSG_DATA  OUT    VARCHAR2,
  PI_CDRBASEOBJCOLL  IN OUT    CDR_BASE_OBJ_COLL,
  PI_CDRTARGETCONTAINEROBJECT  IN OUT    CDR_BASE_OBJ_TYPE,
  PI_CHECKINFLAG  IN    VARCHAR2

);
```

**Parameters**

This API has standard (see "Standard Parameters" on page 5) as well as the following parameters:

- **PI_CDRBASEOBJCOLL** (Mandatory). This is a collection of CDR_BASE_OBJ_TYPES. For each object that you want to copy into the Application Area, initialize a CDR_BASE_OBJ_TYPE and then extend the collection.

The following attributes are required: COMPANY_ID, OBJ_ID, OBJ_VER, OBJECT_VERSION_NUMBER, NAMESPACE_OBJ_ID, NAMESPACE_OBJ_VER.

- **PI_CDRTARGETCONTAINEROBJECT** (Mandatory). This is a parameter of table type CDR_BASE_OBJ_TYPE that contains object attributes. Enter values to identify the Application Area into which you want to copy objects.

The following attributes are required: COMPANY_ID, OBJECT_ID, OBJECT_VER, NAMESPACE_OBJ_ID, NAMESPACE_OBJ_VER, OBJECT_VERSION_NUMBER.

- **PI_CHECKINFLAG** (Mandatory). Enter $YESNO$NO if you want any checked-out copied objects to remain checked out, or $YESNO$YES if you want the system to check them in after the copy operation.

### 3.1.5 Move Objects into an Application Area

Use this API to move LSH objects into an Application Area.

**Name**  
CDR_PUB_DF_APPLICATIONAREA.MoveObjectIntoAA

**Signature**

```
PROCEDURE MOVOBJECTINTOAA(
  P_API_VERSION  IN    NUMBER,
  P_INIT_MSG_LIST  IN    VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_COMMIT  IN    VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_VALIDATION_LEVEL  IN    NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
  X_RETURN_STATUS  OUT    VARCHAR2,
  X_MSG_COUNT  OUT    NUMBER,
  X_MSG_DATA  OUT    VARCHAR2,
  PI_CDRBASEOBJCOLL  IN OUT    CDR_BASE_OBJ_COLL,
  PI_CDRTARGETCONTAINEROBJECT  IN OUT    CDR_BASE_OBJ_TYPE,
  PI_CHECKINFLAG  IN    VARCHAR2

);
```

**Parameters**

This API has standard (see "Standard Parameters" on page 5) as well as the following parameters:

- **PI_CDRBASEOBJCOLL** (Mandatory). This is a collection of CDR_BASE_OBJ_TYPES. For each object that you want to move (including all its contents), initialize a CDR_BASE_OBJ_TYPE and then extend the collection.

The following attributes are required: COMPANY_ID, OBJ_ID, OBJ_VER, OBJECT_VERSION_NUMBER, NAMESPACE_OBJ_ID, NAMESPACE_OBJ_VER.
**Signature**

```sql
PROCEDURE MOVEOBJECTINTOAA(
    P_API_VERSION IN NUMBER,
    P_INIT_MSG_LIST IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_COMMIT IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_VALIDATION_LEVEL IN NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
    X_RETURN_STATUS OUT VARCHAR2,
    X_MSG_COUNT OUT NUMBER,
    X_MSG_DATA OUT VARCHAR2,
    PI_CDRBASEOBJCOLL IN OUT CDR_BASE_OBJ_COLL,
    PI_CDRTARGETCONTAINEROBJECT IN OUT CDR_BASE_OBJ_TYPE
);
```

**Parameters**  This API has standard (see "Standard Parameters" on page 5) as well as the following parameters:

- **PI_CDRBASEOBJCOLL** (Mandatory) This is a collection of CDR_BASE_OBJ_TYPES. For each object that you want to move into the Application Area, initialize a CDR_BASE_OBJ_TYPE and then extend the collection.

  The following attributes are required: COMPANY_ID, OBJ_ID, OBJ_VER, OBJECT_VERSION_NUMBER, NAMESPACE_OBJ_ID, NAMESPACE_OBJ_VER.

- **PI_CDRTARGETCONTAINEROBJECT** (Mandatory) This is a parameter of table type CDR_BASE_OBJ_TYPE that contains object attributes. Enter values to identify the Application Area into which you want to move objects.

  The following attributes are required: COMPANY_ID, OBJ_ID, OBJ_VER, OBJECT_VERSION_NUMBER, NAMESPACE_OBJ_ID, NAMESPACE_OBJ_VER.
This section contains the following topics:

- Section 4.1, "Define and Modify Business Areas"
- Section 4.2, "Create and Modify Business Area Hierarchies"
- Section 4.3, "Create and Modify Business Area Joins"
- Section 4.4, "Public APIs and Naming Views for Generic Visualization Adapter (GVA)"

4.1 Define and Modify Business Areas

This is a public interface for Business Area-related APIs, including creating, modifying, and removing Business Areas. It also includes APIs for checking Business Areas in and out, and undoing a Business Area checkout.

This section contains the following topics:

- Section 4.1.1, "Create a Business Area"
- Section 4.1.2, "Modify a Business Area"
- Section 4.1.3, "Check Out a Business Area"
- Section 4.1.4, "Undo a Business Area Checkout"
- Section 4.1.5, "Check In a Business Area"
- Section 4.1.6, "Remove a Business Area"

4.1.1 Create a Business Area

Use this API to create a new Business definition, a new instance of an existing Business Area definition, or a new definition and an instance of it.

Name  CDR_PUB_DF_BUSINESSAREA.CreateBusiArea

Signature

PROCEDURE CREATEBUSIAREA(
    P_API_VERSION IN    NUMBER,
    P_INIT_MSG_LIST IN    VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_COMMIT IN    VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_VALIDATION_LEVEL IN    NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
    X_RETURN_STATUS OUT    VARCHAR2,
    X_MSG_COUNT OUT    NUMBER,
    X_MSG_DATA OUT    VARCHAR2,
Define and Modify Business Areas

Parameters  This API has standard parameters (see "Standard Parameters" on page 5) and the following parameters:

- PIO_SOURCECDRNAMING (Mandatory) This is a parameter of table type CDR_NAMING_VERSIONS_OBJ_TYPE that contains CDR Naming Version attributes.
  
  If you are creating a new definition only or a new definition and an instance of it, enter values for the new definition.

  If you are creating an instance of an existing definition, enter values to identify the definition. For OBJECT_TYPE_RC enter $OBJTYPES$BUSAREA if you are creating a definition only; $OBJTYPES$BUSAREAREF if you are creating an instance of an existing definition; and NULL if you are creating a new definition and an instance of it.

- PI_CDRPREGOBJTYPE (Optional) This is a parameter of table type CDR_BUSAREA_OBJ_TYPE that contains object attributes specific to Business Areas.

  If you are creating a new definition, enter values for the new Business Area.

  The following attributes are required: ADAPTER_COMPANY_ID,ADAPTER_OBJ_ID,ADAPTER_OBJ_VER.

  If you are creating an instance of an existing Business Area, do not enter any values here.

- PI_CREATEOBJECT (Mandatory) Enter DEFN to create a definition only; INST to create a instance of an existing definition; or BOTH to create a new definition and an instance of it.

- PI_INSTANCE_SUBTYPE_ID (Optional) If you are creating a new instance, enter the ID for the subtype you want to give the instance.

  If you are creating a definition only, do not enter a value for this parameter.

- PI_DEFCLASSIFICATIONCOLL (Optional) By default, the new definition is classified according to the subtype you assigned it in the CDR_NAMING_VERSION_OBJ_TYPE.

  If you want to override the default classifications for one or more classification levels, use this parameter. This is a collection of CDR_CLA_OBJ_TYPES, which have 5 attributes, including CLA_LEVEL_ID and CLASSIFICATION_ID.

  If you want the definition to inherit its classifications for a particular level from its parent, enter the classification level ID and, for the CLASSIFICATION_ID, enter 0 (zero).

  If you want to explicitly assign one or more terms for a particular level, initialize a CDR_CLA_OBJ_TYPE for each term, entering the classification level ID and, for the CLASSIFICATION_ID, the term ID. The PAR_ attributes are not relevant to Business Areas. Do not enter any values for them.
PI_INSTCLASSIFICATIONCOLL (Optional) By default, the new instance is classified according to the subtype you assigned it in the PI_INSTANCE_SUBTYPE_ID.

If you want to override the default classifications for one or more classification levels, use this parameter. This is a collection of CDR_CLA_OBJ_TYPES, which have 5 attributes, including CLA_LEVEL_ID and CLASSIFICATION_ID.

If you want the instance to inherit its classifications for a particular level from its parent Work Area, enter the classification level ID and, for the CLASSIFICATION_ID, enter 0 (zero).

If you want to explicitly assign one or more terms for a particular level, initialize a CDR_CLA_OBJ_TYPE for each term, entering the classification level ID and, for the CLASSIFICATION_ID, the term ID. The PAR_attributes are not relevant to Business Areas. Do not enter any values for them.

4.1.2 Modify a Business Area

Use this API to modify a Business Area definition or instance.

**Note:** If you are modifying a definition, you must first check it out.

**Name**  CDR_PUB_DF_BUSINESSAREA.ModifyBusiAreaDetails

**Signature**

PROCEDURE MODIFYBUSIAREADETAILS(
P_API_VERSION  IN    NUMBER,
P_INIT_MSG_LIST  IN    VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
P_COMMIT  IN    VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
P_VALIDATION_LEVEL  IN    NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
X_RETURN_STATUS  OUT    VARCHAR2,
X_MSG_COUNT  OUT    NUMBER,
X_MSG_DATA  OUT    VARCHAR2,
PIO_CDRNAMING  IN OUT    CDR_NAMING_VERSION_OBJ_TYPE
);

**Parameters**  This API has standard parameters (see "Standard Parameters" on page 5) and the following parameter:

PIO_CDRNAMING (Mandatory) This is a parameter of table type CDR_NAMING_VERSION_OBJ_TYPE that contains object attributes. Enter values to identify the Business Area and enter new values for the attributes you want to modify.

You can change the name, description, or version label for either a definition or instance. For an instance, you can also change to a different underlying source definition by entering values for the new definition in the three REF attributes. All attributes are required.

4.1.3 Check Out a Business Area

Use this API to check out a Business Area definition either directly or through an instance of it.

**Name**  CDR_PUB_DF_BUSINESSAREA.CheckOutBusiArea

**Signature**
PROCEDURE CHECKOUTBUSIAREA(  
  P_API_VERSION IN NUMBER,  
  P_INIT_MSG_LIST IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,  
  P_COMMIT IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,  
  P_VALIDATION_LEVEL IN NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,  
  X_RETURN_STATUS OUT VARCHAR2,  
  X_MSG_COUNT OUT NUMBER,  
  X_MSG_DATA OUT VARCHAR2,  
  PIO_BASEOBJECT IN OUT CDR_BASE_OBJ_TYPE,  
  PI_COMMENT IN VARCHAR2,  
  PI_ISINSTONLY IN VARCHAR2  
);  

Parameters This API has standard parameters (see “Standard Parameters” on page 5) and the following parameters:

■ PIO_BASEOBJECT (Mandatory) This is a parameter of table type CDR_BASE_OBJ_TYPE that contains object attributes. Enter values to identify the Business Area that you want to do check out.

  If you are checking out the Business Area definition directly, enter values to identify the definition.

  If you are checking out a Business Area definition through an instance of it, enter values to identify the instance.

  The following attributes are required: COMPANY_ID,OBJ_ID,OBJ_VER,OBJECT_VERSION_NUMBER,NAMESPACE_OBJ_ID,NAMESPACE_OBJ_VER.

■ PI_COMMENT Enter the reason you are checking out the Business Area.

■ PI_ISINSTONLY (Mandatory) Enter $YESNO$NO.

4.1.4 Undo a Business Area Checkout

Use this API to undo the checkout of a Business Area definition, discarding any changes that have been made.

Name CDR_PUB_DF_BUSINESSAREA.UncheckOutBusiArea

Signature

PROCEDURE UNCHECKOUTBUSIAREA(  
  P_API_VERSION IN NUMBER,  
  P_INIT_MSG_LIST IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,  
  P_COMMIT IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,  
  P_VALIDATION_LEVEL IN NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,  
  X_RETURN_STATUS OUT VARCHAR2,  
  X_MSG_COUNT OUT NUMBER,  
  X_MSG_DATA OUT VARCHAR2,  
  PIO_BASEOBJECT IN OUT CDR_BASE_OBJ_TYPE  
);  

Parameters This API has standard parameters (see “Standard Parameters” on page 5) and the following parameters:

PIO_BASEOBJECT (Mandatory) This is a parameter of table type CDR_BASE_OBJ_TYPE that contains object attributes. Enter values to identify the Business Area whose checkout you want to undo.

  The following attributes are required: COMPANY_ID,OBJ_ID,OBJ_VER,OBJECT_VERSION_NUMBER,NAMESPACE_OBJ_ID,NAMESPACE_OBJ_VER.
4.1.5 Check In a Business Area

Use this API to check in a Business Area definition.

Name  CDR_PUB_DF_BUSINESSAREA.CheckInBusiAreaDefinition

Signature

PROCEDURE CHECKINBUSIAREADEFINITION(
    P_API_VERSION IN NUMBER,
    P_INIT_MSG_LIST IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_COMMIT IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_VALIDATION_LEVEL IN NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
    X_RETURN_STATUS OUT VARCHAR2,
    X_MSG_COUNT OUT NUMBER,
    X_MSG_DATA OUT VARCHAR2,
    PIO_BASEOBJECT IN OUT CDR_BASE_OBJ_TYPE,
    PI_COMMENT IN VARCHAR2
);

Parameters  This API has standard parameters (see "Standard Parameters" on page 5) and the following parameters:

- **PIO_BASEOBJECT** (Mandatory) This is a parameter of table type CDR_BASE_OBJ_TYPE that contains object attributes. Enter values to identify the Business Area that you want to check in.
  
  The following attributes are required: COMPANY_ID,OBJ_ID,OBJ_VER,OBJECT_VERSION_NUMBER,NAMESPACE_OBJ_ID,NAMESPACE_OBJ_VER.

- **PI_COMMENT** (Optional) Enter the reason you are checking in the Business Area.

4.1.6 Remove a Business Area

Use this API to remove one or more Business Area definitions or instances.

Name  CDR_PUB_DF_BUSINESSAREA.RemoveBusiArea

Signature

PROCEDURE REMOVEBUSIAREA(
    P_API_VERSION IN NUMBER,
    P_INIT_MSG_LIST IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_COMMIT IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_VALIDATION_LEVEL IN NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
    X_RETURN_STATUS OUT VARCHAR2,
    X_MSG_COUNT OUT NUMBER,
    X_MSG_DATA OUT VARCHAR2,
    PIO_CDRBASEOBJCOLL IN OUT CDR_BASE_OBJ_COLL
);

Parameters  This API has standard parameters (see "Standard Parameters" on page 5) and the following parameter:

- **PIO_CDRBASEOBJCOLL** (Mandatory) This is a collection of CDR_BASE_OBJ_TYPES.

  For each Business Area that you want to remove, initialize a CDR_BASE_OBJ_TYPE and then extend the collection.
Create and Modify Business Area Hierarchies

This is a public interface for all Business Area hierarchy-related operations, including creating, modifying, and removing Business Area hierarchies and hierarchy columns.

This section contains the following topics:

- Section 4.2.1, "Create a Business Area Hierarchy"
- Section 4.2.2, "Modify a Hierarchy and a Hierarchy Column"
- Section 4.2.3, "Reorder a Hierarchy Column"
- Section 4.2.4, "Remove a Hierarchy Column"
- Section 4.2.5, "Remove a Business Area Hierarchy"

4.2.1 Create a Business Area Hierarchy

Use this API to create a hierarchy in a Business Area. To define the hierarchy's columns, use ModifyBusAreaHierAndHierCol.

Name  CDR_PUB_DF_BUSINESSAREA_HIER.CreateBusinessAreaHier

Signature

PROCEDURE CREATEBUSINESSAREAHIER(
  P_API_VERSION IN NUMBER,
  P_INIT_MSG_LIST IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_COMMIT IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_VALIDATION_LEVEL IN NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
  X_RETURN_STATUS OUT VARCHAR2,
  X_MSG_COUNT OUT NUMBER,
  X_MSG_DATA OUT VARCHAR2,
  PIO_SOURCECDRNAMING IN OUT CDR_NAMING_VERSION_OBJ_TYPE
);

Parameters  This API has standard parameters (see "Standard Parameters" on page 5) and the following parameters:

PIO_SOURCECDRNAMING (Mandatory) This is a parameter of table type CDR_NAMING_VERSION_OBJ_TYPE that contains object attributes.

Enter values for the new hierarchy. Use the NAMESPACE attributes to identify the Business Area in which you want to create the hierarchy.

The following attributes are required: OBJECT_TYPE_RC, NAME, NAMESPACE_OBJ_ID, NAMESPACE_OBJ_VER, OBJECT_SUBTYPE_ID. For OBJECT_TYPE_RC enter $OBJTYPE$BUSAREAHIER.

4.2.2 Modify a Hierarchy and a Hierarchy Column

Use this API to modify Business Area Hierarchies and Hierarchy Columns. You can change name and description, add Columns, and change their Group With Previous setting. Use the Remove Columns API to remove Columns.

Name  CDR_PUB_DF_BUSINESSAREA_HIER.ModifyBusAreaHierAndHierCol
Create and Modify Business Area Hierarchies

**Signature**

```sql
PROCEDURE MODIFYBUSAREAHIERANDHIERCOL(
  P_API_VERSION  IN    NUMBER,
  P_INIT_MSG_LIST  IN    VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_COMMIT  IN    VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_VALIDATION_LEVEL  IN    NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
  X_RETURN_STATUS  OUT    VARCHAR2,
  X_MSG_COUNT  OUT    NUMBER,
  X_MSG_DATA  OUT    VARCHAR2,
  PIO_NAMING  IN OUT    CDR_NAMING_VERSION_OBJ_TYPE,
  PI_BUS_HIERCOLUMNS_COLL  IN OUT    CDR_BUSAREA_HIER_COLS_COLL
);
```

**Parameters**

This API has standard parameters (see “Standard Parameters” on page 5) and the following parameters:

- **PIO_NAMING** (Mandatory) This is a parameter of table type CDR_NAMING_VERSION_OBJ_TYPE that contains object attributes.
  
  Enter values to identify the Business Area Hierarchy that you want to modify and enter new values for the attributes you want to modify. You can modify only the Name and Description. All attributes are required.

- **PI_BUS_HIERCOLUMNS_COLL** (Mandatory) This is a collection of CDR_BUSAREA_HIER_COLS_OBJ_TYPES.
  
  Initialize a CDR_BUSAREA_HIER_COLS_OBJ_TYPE for each Column in their position order with the values you want to change, and then extend the collection. All attributes are required.

**4.2.3 Reorder a Hierarchy Column**

Use this API to reorder the Columns of a Business Area Hierarchy.

---

**Note:** The API enforces validation rules for Column sequence.

---

**Name**

CDR_PUB_DF_BUSINESSAREA_HIER.ReorderHierCols

**Signature**

```sql
PROCEDURE REORDERHIERCOLS(
  P_API_VERSION  IN    NUMBER,
  P_INIT_MSG_LIST  IN    VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_COMMIT  IN    VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_VALIDATION_LEVEL  IN    NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
  X_RETURN_STATUS  OUT    VARCHAR2,
  X_MSG_COUNT  OUT    NUMBER,
  X_MSG_DATA  OUT    VARCHAR2,
  PIO_BUS_HIERCOLUMNS_COLL  IN OUT    CDR_BUSAREA_HIER_COLS_COLL
);
```

**Parameters**

This API has standard parameters (see “Standard Parameters” on page 5) and the following parameters:

- **PIO_BUS_HIERCOLUMNS_COLL** (Mandatory) This is a collection of CDR_BUSAREA_HIER_COLS_OBJ_TYPES. For each Column, initialize a CDR_BUSAREA_HIER_COLS_OBJ_TYPE, providing the new value for POSITION NUMBER relative to
the other Columns, and then extend the collection. You must initialize all existing Columns.

The following attributes are required: `COMPANY_ID,HIER_OBJ_ID,HIER_OBJ_VER,TD_COMPANY_ID,TD_OBJ_ID,TD_COL_COMPANY_ID,TD_COL_OBJ_ID,POSITION,GROUP_WITH_PREVIOUS_RC`.

### 4.2.4 Remove a Hierarchy Column

Use this API to remove Business Area Hierarchy Columns.

**Note:** You cannot remove a Column that is currently part of a Join. If you remove a Column, and if the next Column’s Group By Previous setting is Yes, then that setting is changed to No.

**Name**  
CDR_PUB_DF_BUSINESSAREA_HIER.RemoveHierColumns

**Signature**

```sql
PROCEDURE REMOVEHIERCOLUMNS(
    P_API_VERSION IN NUMBER,
    P_INIT_MSG_LIST IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_COMMIT IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_VALIDATION_LEVEL IN NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
    X_RETURN_STATUS OUT VARCHAR2,
    X_MSG_COUNT OUT NUMBER,
    X_MSG_DATA OUT VARCHAR2,
    PIO_BUS_HIERCOLUMNS_COLL IN OUT CDR_BUSAREA_HIER_COLS_COLL
);
```

**Parameters**  
This API has standard parameters (see "Standard Parameters" on page 5) and the following parameter:

**PIO_BUS_HIERCOLUMNS_COLL** (Mandatory) This is a collection of `CDR_BUSAREA_HIER_COLS_OBJ_TYPE`s. For each Column that you want to remove, initialize a `CDR_BUSAREA_HIER_COLS_OBJ_TYPE` and then extend the collection.

The following attributes are required: `COMPANY_ID,HIER_OBJ_ID,HIER_OBJ_VER,TD_COMPANY_ID,TD_OBJ_ID,TD_COL_COMPANY_ID,TD_COL_OBJ_ID,POSITION,GROUP_WITH_PREVIOUS_RC`.

### 4.2.5 Remove a Business Area Hierarchy

Use this API to remove one or more Business Area Hierarchies, including all their Columns.

**Name**  
CDR_PUB_DF_BUSINESSAREA_HIER.RemoveHiers

**Signature**

```sql
PROCEDURE REMOVEHIERS(
    P_API_VERSION IN NUMBER,
    P_INIT_MSG_LIST IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_COMMIT IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_VALIDATION_LEVEL IN NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
    X_RETURN_STATUS OUT VARCHAR2,
    X_MSG_COUNT OUT NUMBER,
    X_MSG_DATA OUT VARCHAR2,
    PIO_BUS_HIERCOLUMNS_COLL IN OUT CDR_BUSAREA_HIER_COLS_COLL
);
```
Create and Modify Business Area Joins

This API has standard parameters (see “Standard Parameters” on page 5) and the following parameter:

**PI_BASEOBJCOLL** (Mandatory) This is a collection of CDR_BASE_OBJ_TYPEs. For each Hierarchy that you want to remove, initialize a CDR_BASE_OBJ_TYPE and then extend the collection.

The following attributes are required: COMPANY_ID,OBJ_ID,OBJ_VER,OBJECT_VERSION_NUMBER,NAMESPACE_OBJ_ID,NAMESPACE_OBJ_VER.

4.3 Create and Modify Business Area Joins

This is a public interface for Business Area Join-related operations, including creating, modifying, and removing Business Area Joins and Join Columns.

This section contains the following topics:

- **Section 4.3.1, "Create a Join"**
- **Section 4.3.2, "Modify a Join and a Join Column"**
- **Section 4.3.3, "Remove a Join Column"**
- **Section 4.3.4, "Remove a Join"**

4.3.1 Create a Join

Use this API to create a Business Area Join.

**Name**  
CDR_PUB_DF_BUSINESSAREA_JOIN.CreateBusinessAreaJoin

**Signature**

PROCEDURE CREATEBUSINESSAREAJOIN(
  P_API_VERSION IN NUMBER,
  P_INIT_MSG_LIST IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_COMMIT IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_VALIDATION_LEVEL IN NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
  X_RETURN_STATUS OUT VARCHAR2,
  X_MSG_COUNT OUT NUMBER,
  X_MSG_DATA OUT VARCHAR2,
  PIO_SOURCECDRNAMING IN OUT CDR_NAMING_VERSION_OBJ_TYPE,
  PI_BUSAREAJOINSOBJTYPE IN CDR_BUSAREA_JOINS_OBJ_TYPE
);

**Parameters**  
This API has standard parameters (see “Standard Parameters” on page 5) and the following parameters:

- **PIO_SOURCECDRNAMING** (Mandatory) This is a parameter of table type CDR_NAMING_VERSION_OBJ_TYPE that contains object attributes. Enter values for the Join that you want to create. Use the NAMESPACE attributes to identify the Business Area in which you want to create the Join.

  The following attributes are required: OBJECT_TYPE_RC,NAME,NAMESPACE_OBJ_ID,NAMESPACE_OBJ_VER,OBJECT_SUBTYPE_ID. For OBJECT_TYPE_RC enter $OBJTYPES$JOIN.
Create and Modify Business Area Joins

- **PI_BUSAREAJOINSOBJTYPE** (Mandatory) This is a parameter of table type CDR_BUSAREA_JOINS_OBJ_TYPE that contains object attributes specific to Joins. Enter values for the Join that you want to create.

  The following attributes are required: TD_COMPANY_ID, TD_OBJ_ID, FK_TD_COMPANY_ID, FK_TD_OBJ_ID, TD_OUTERJOIN_RC, FK_TD_OUTERJOIN_RC. For TD_OUTERJOIN_RC and FK_TD_OUTERJOIN_RC enter $YESNO$NO to define an inner join on the side of the corresponding Table Descriptor, or $YESNO$YES to define an outer join. Be sure to define an outer join on only one side, if any.

4.3.2 Modify a Join and a Join Column

Use this API to modify Business Area Joins and Join Columns. You can change the name and description of the Join and change either side to an inner or outer join (but be careful not to make both sides into outer joins). You can add Join Columns and pair them with a different Column from the other Table Descriptor.

**Note:** Do not change the operator. EQUAL TO is the only operator currently supported.

Name CDR_PUB_DF_BUSINESSAREA_JOIN.ModifyBusAreaJoinAndJoinCol

Signature

```java
PROCEDURE MODIFYBUSAREAJOINANDJOINCOL(
    P_API_VERSION IN NUMBER,
    P_INIT_MSG_LIST IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_COMMIT IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_VALIDATION_LEVEL IN NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
    X_RETURN_STATUS OUT VARCHAR2,
    X_MSG_COUNT OUT NUMBER,
    X_MSG_DATA OUT VARCHAR2,
    PIO_NAMING IN OUT CDR_NAMING_VERSION_OBJ_TYPE,
    PI_BUSINESS_JOIN IN OUT CDR_BUSAREA_JOINS_OBJ_TYPE,
    PI_BUS_JOINCOLUMNS_COLL IN OUT CDR_BUSAREA_JOIN_COLS_COLL
);
```

Parameters This API has standard parameters (see “Standard Parameters” on page 5) and the following parameters:

- **PIO_NAMING** (Mandatory) This is a parameter of table type CDR_NAMING_VERSION_OBJ_TYPE that contains object attributes. Enter values to identify the Join that you want to modify and for the attributes you want to change.

  The following attributes are required: COMPANY_ID, OBJ_ID, OBJ_VER, OBJECT_VERSION_NUMBER, NAMESPACE_OBJ_ID, NAMESPACE_OBJ_VER.

- **PI_BUSINESS_JOIN** (Mandatory) This is a parameter of table type CDR_BUSAREA_JOINS_OBJ_TYPE that contains object attributes specific to Joins. Enter values to identify the Join that you want to modify and the values you want to modify. All attributes are required.

- **PI_BUS_JOINCOLUMNS_COLL** (Mandatory) This is a collection of CDR_BUSAREA_JOIN_COLS_TYPEs. For each Join Column that you want to modify, initialize a CDR_BUSAREA_JOIN_COLS_TYPE and then extend the collection.

  The following attributes are required: COMPANY_ID, JOIN_OBJ_ID, JOIN_OBJ_VER, TD_COL_COMPANY_ID, TD_COL_OBJ_ID, FK_TD_COL_COMPANY_
Create and Modify Business Area Joins

ID, FK_TD_COL_OBJ_ID, POSITION. You can change the joined columns but you cannot modify the operator, which is always Equal To.

4.3.3 Remove a Join Column

Use this API to remove one or more Columns from a Join.

Name  CDR_PUB_DF_BUSINESSAREA_JOIN.RemoveJoinColumns

Signature

PROCEDURE REMOVEJOINCOLUMNS(
    P_API_VERSION  IN    NUMBER,
    P_INIT_MSG_LIST  IN    VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_COMMIT  IN    VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_VALIDATION_LEVEL  IN    NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
    X_RETURN_STATUS  OUT    VARCHAR2,
    X_MSG_COUNT  OUT    NUMBER,
    X_MSG_DATA  OUT    VARCHAR2,
    PIO_BUS_JOINCOLUMNS_COLL  IN OUT    CDR_BUSAREA_JOIN_COLS_COLL
);  

Parameters

This API has standard parameters (see “Standard Parameters” on page 5) and the following parameter:

PIO_BUS_JOINCOLUMNS_COLL (Mandatory) This is a collection of CDR_BUSAREA_JOIN_COLS_TYPEs. For each Join Column that you want to remove, initialize a CDR_BUSAREA_JOIN_COLS_TYPE and then extend the collection.

The following attributes are required: COMPANY_ID, JOIN_OBJ_ID, JOIN_OBJ_VER, TD_COL_COMPANY_ID, TD_COL_OBJ_ID, FK_TD_COL_COMPANY_ID, FK_TD_COL_OBJ_ID, POSITION.

4.3.4 Remove a Join

Use this API to remove one or more Joins, with all their Columns, from a Business Area.

Name  CDR_PUB_DF_BUSINESSAREA_JOIN.RemoveJoins

Signature

PROCEDURE REMOVEJOINS(
    P_API_VERSION  IN    NUMBER,
    P_INIT_MSG_LIST  IN    VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_COMMIT  IN    VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_VALIDATION_LEVEL  IN    NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
    X_RETURN_STATUS  OUT    VARCHAR2,
    X_MSG_COUNT  OUT    NUMBER,
    X_MSG_DATA  OUT    VARCHAR2,
    PI_BASEOBJ_COLL  IN OUT    CDR_BASE_OBJ_COLL
);  

Parameters

This API has standard parameters (see “Standard Parameters” on page 5) and the following parameter:

PI_BASEOBJ_COLL (Mandatory) This is a collection of CDR_BASE_OBJ_TYPEs.

For each Join that you want to remove from the Business Area, initialize a CDR_BASE_OBJ_TYPE and then extend the collection.
The following attributes are required: COMPANY_ID,OBJ_ID,OBJ_VER,OBJECT_VERSION_NUMBER,NAMESPACE_OBJ_ID,NAMESPACE_OBJ_VER.

4.4 Public APIs and Naming Views for Generic Visualization Adapter (GVA)

This is a public interface for operations related to Generic Visualization Business Areas including initializing and resetting them.

This section contains the following topics:

- Section 4.4.2, "Reset a Generic Visualization Business Area"
- Section 4.4.3, "Get Possible Blinding Types of a Business Area Instance"
- Section 4.4.4, "Get Snapshot Labels Common to all Tables in a BA Instance for a Given Blinding AccessType"
- Section 4.4.5, "Naming Views"

4.4.1 Initialize a Generic Visualization Business Area Instance

Use this API to initialize a particular Generic Visualization Business Area Instance of a given currency and blinding access type. If the currency and the blinding access type values are null, the default types of the BA instance are used.

You can invoke this API multiple times in the same instance in a single user session to change the type of access to the data. You can also invoke it multiple times to read data from more than one Business Area Schema as long as all the data across all the Business Areas is uniformly either real or dummy.

If you initialize a new Business Area schema changes blinding access, the API errors out, directing you to reset access to all Business Areas using the RESETBAACCESS API.

**Name**  CDR_PUB_API_GVA.SetInitializeBA

**Signature**

PROCEDURE SETINITIALIZEBA(
    PI_COMPANYID IN CDR_NAMINGS.COMPANY_ID%TYPE,
    PI_OBJID     IN CDR_NAMINGS.OBJ_ID%TYPE,
    PI_OBJVER    IN CDR_NAMING_VERSIONS.OBJ_VER%TYPE,
    PI_VCURRENCY IN VARCHAR2,
    PI_VBLINDINGACCESSTYPE IN VARCHAR2,
    X_RETURN_STATUS OUT NOCOPY VARCHAR2,
    X_MSG_COUNT OUT NOCOPY NUMBER,
    X_MSG_DATA OUT NOCOPY VARCHAR2
);

**Parameters**

This API has the following parameters:

- **PI_COMPANYID.** Enter the Business Area Instance Company ID.
- **PI_OBJID.** Enter the Business Area Instance Object ID.
- **PI_OBJVER.** Enter the Business Area Instance Object Version.
- **PI_VCURRENCY.** Enter the Currency status.
- **PI_VBLINDINGACCESSTYPE.** Enter the Blinding Access Type.
4.4.2 Reset a Generic Visualization Business Area

Use this API to clear all the initializations of Business Area schemas. It is equivalent to logging out and logging back in to the system. Use this API when you want to change blinding access type from blinded to unblinded or vice-versa.

**Name**  
CDR_PUB_API_GVA.RESETBAACCESS

4.4.3 Get Possible Blinding Types of a Business Area Instance

Use this API to get the possible Blinding Access Types of a Business Area Instance, which is in turn based on the blinding statuses of underlying Business Area Table instances and the user’s privileges.

**Name**  
CDR_PUB_API_GVA.GetBAValidBlindingAccessTypes

**Signature**

FUNCTION GETBAVALIDBLINDINGACCESSTYPES(  
PI_COMPANYID IN CDR_NAMINGS.COMPANY_ID%TYPE,  
PI_OBJID     IN CDR_NAMINGS.OBJ_ID%TYPE,  
PI_OBJVER    IN CDR_NAMING_VERSIONS.OBJ_VER%TYPE) RETURN BLINDINGACCESSTYPESCOLL  
PIPELINED;

**Return**  
A collection of the possible blinding access types.  
BLINDINGACCESSTYPESCOLL is TABLE TYPE OF VARCHAR2(30);

**Parameters**  
This API has the following parameters:

- **PI_COMPANYID.** Enter the Business Area Instance Company ID.
- **PI_OBJID.** Enter the Business Area Instance Object ID.
- **PI_OBJVER.** Enter the Business Area Instance Object Version.
- **BLINDINGACCESSTYPESCOLL.** This is the list of possible blinding access types.

4.4.4 Get Snapshot Labels Common to all Tables in a BA Instance for a Given Blinding AccessType

Use this API to get the snapshot labels common to all Tables within a Business Area Instance for a given blinding access type.

**Name**  
CDR_PUB_API_GVA.GetSnapshotLabels

**Signature**

FUNCTION GETSNAPSHOTLABELS(  
PI_COMPANYID IN CDR_NAMINGS.COMPANY_ID%TYPE,  
PI_OBJID     IN CDR_NAMINGS.OBJ_ID%TYPE,  
PI_OBJVER    IN CDR_NAMING_VERSIONS.OBJ_VER%TYPE) RETURN VARCHAR2;

**Parameters**  
This API has the following parameters:

- **PI_COMPANYID.** Enter the Business Area Instance Company ID.
- **PI_OBJID.** Enter the Business Area Instance Object ID.
- **PI_OBJVER.** Enter the Business Area Instance Object Version.
```sql
PI_OBJVER IN CDR_NAMING_VERSIONS.OBJ_VER%TYPE,
PI_VBLINDINGACCESTYPE IN VARCHAR2) RETURN CURRENCYCOLL PIPELINED;
```

**Return**  A collection of the snapshot labels for a particular blinding access type in the Business Area instance common to all Tables within a Business Area Instance. CURRENCYCOLL is TABLE TYPE OF VARCHAR2(200);

**Parameters**  This API has the following parameters.

- **PI_COMPANYID**. Enter the Business Area Instance Company ID.
- **PI_OBJID**. Enter the Business Area Instance Object ID.
- **PI_OBJVER**. Enter the Business Area Instance Object Version.
- **PI_VBLINDINGACCESTYPE**. Enter the Blinding Access Type.

### 4.4.5 Naming Views

**CDR_PUB GENERIC BA V**

Use this naming view to retrieve all the Generic Visualization Business Area Instances on which a user has privileges to read data.

**CDR_PUB GENERIC BA TABLES_V**  Use this view to retrieve the Table instance details for a given GV BAI.
This is a public interface for creating, modifying, and removing Data Marts.

5.1 Define and Modify Work Areas

This section contains the following topics:

- Section 5.1.1, "Create a Data Mart"
- Section 5.1.2, "Check In a DataMart Definition"
- Section 5.1.3, "Modify a Data Mart"
- Section 5.1.4, "Check Out a Data Mart"
- Section 5.1.5, "Remove a Data Mart"

5.1.1 Create a Data Mart

Use this API to create a new Data Mart definition, a new instance of an existing Data Mart definition, or a new definition and an instance of it.

**Name**  
CDR_PUB_DF_DATAMART.CreateDataMart

**Signature**

```sql
PROCEDURE CREATEDATAMART(
    P_API_VERSION IN NUMBER,
    P_INIT_MSG_LIST IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_COMMIT IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_VALIDATION_LEVEL IN NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
    X_RETURN_STATUS OUT VARCHAR2,
    X_MSG_COUNT OUT NUMBER,
    X_MSG_DATA OUT VARCHAR2,
    PIO_SOURCECDRNAMING IN OUT CDR_NAMING_VERSION_OBJ_TYPE,
    PI_CORDATAMARTOBJTYPE IN CDR_DATA_MART_OBJ_TYPE,
    PI_CREATEOBJECT IN VARCHAR2,
    PI_INSTANCE_SUBTYPE_ID IN CDR_NAMINGS.OBJECT_SUBTYPE_ID%TYPE,
    PI_DEFCLASSIFICATIONCOLL IN CDR_CLASSIFICATIONS_COLL,
    PI_INSTCLASSIFICATIONCOLL IN CDR_CLASSIFICATIONS_COLL
);```

**Parameters**  
This API has standard parameters (see "Standard Parameters" on page 5) and the following parameters:
- **PIO_SOURCECDRNAMING** (Mandatory) This is a parameter of table type CDR_NAMING_VERSIONS_OBJ_TYPE that contains CDR Naming Version attributes.

  If you are only creating a new definition or a new definition and an instance of it, enter values for the new definition. If you are creating a definition enter $OBJTYPES$DATAMART for OBJECT_TYPE_RC. Enter NULL if you are creating a new definition and an instance of it.

  If you are creating an instance of an existing definition, enter values to identify the definition. If you are creating an instance of an existing definition enter $OBJTYPES$DATAMARTREF for OBJECT_TYPE_RC.

- **PI_CDRDATAMARTOBJTYPE** (Optional) This is a parameter of table type CDR_DATA_MART_OBJ_TYPE that contains object attributes specific to Data Marts.

  If you are creating a new definition, enter values for the new Data Mart. The following attributes are required: COMPANY_ID, ADAPTER_COMPANY_ID, ADAPTER_OBJ_ID, ADAPTER_OBJ_VER.

  If you are creating an instance of an existing Data Mart, do not enter any values here.

- **PI_CREATEOBJECT** (Mandatory) Enter DEFN to create a definition only; INST to create a instance of an existing definition; or BOTH to create a new definition and an instance of it.

- **PI_INSTANCE_SUBTYPE_ID** (Optional) If you are creating a new instance, enter the ID for the subtype you want to give the instance. If you are creating a definition only, do not enter a value for this parameter.

- **PI_DECLASSIFICATIONCOLL** (Optional) By default the new definition is classified according to the subtype you assigned it in the CDR_NAMING_VERSION_OBJ_TYPE. If you want to override the default classifications for one or more classification levels, use this parameter. This is a collection of CDR_CLA_OBJ_TYPES, which have 5 attributes, including CLA_LEVEL_ID and CLASSIFICATION_ID.

  If you want the definition to inherit its classifications for a particular level from its parent, enter the classification level ID and, for the CLASSIFICATION_ID, enter 0 (zero).

  If you want to explicitly assign one or more terms for a particular level, initialize a CDR_CLA_OBJ_TYPE for each term, entering the classification level ID and, for the CLASSIFICATION_ID, the term ID. The PAR_ attributes are not relevant to Data Marts. Do not enter any values for them. If you are not creating a new definition, do not enter values here.

- **PI_INSTCLASSIFICATIONCOLL** The definition is classified according to the subtype you assigned it in the PI_INSTANCE_SUBTYPE_ID. If you want to override the default classifications for one or more classification levels, use this parameter. This is a collection of CDR_CLA_OBJ_TYPES, which have 5 attributes, including CLA_LEVEL_ID and CLASSIFICATION_ID. If you want the instance to inherit its classifications for a particular level from its parent Work Area, enter the classification level ID and, for the CLASSIFICATION_ID, enter 0 (zero). If you want to explicitly assign one or more terms for a particular level, initialize a CDR_CLA_OBJ_TYPE for each term, entering the classification level ID and, for the CLASSIFICATION_ID, the term ID. The PAR_ attributes are not relevant to Data Marts. Do not enter any values for them. If you are not creating a new instance, do not enter values here.
5.1.2 Check In a DataMart Definition

Use this API to check in a Data Mart definition.

**Name**  CDR_PUB_DF_DATAMART.CheckInDataMartDefinition

**Signature**

PROCEDURE CHECKINDATAMARTDEFINITION(
  P_API_VERSION  IN    NUMBER,
  P_INIT_MSG_LIST  IN    VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_COMMIT  IN    VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_VALIDATION_LEVEL  IN    NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
  X_RETURN_STATUS  OUT    VARCHAR2,
  X_MSG_COUNT  OUT    NUMBER,
  X_MSG_DATA  OUT    VARCHAR2,
  PIO_BASEOBJECT  IN OUT    CDR_BASE_OBJ_TYPE,
  PI_COMMENT  IN    VARCHAR2
);

**Parameters**  This API has standard parameters (see "Standard Parameters" on page 5) and the following parameters:

- **PIO_BASEOBJECT** (Mandatory) This is a parameter of table type CDR_BASE_OBJ_TYPE that contains object attributes. Enter values to identify the Data Mart definition that you want to check in. The following attributes are required: COMPANY_ID,OBJ_ID,OBJ_VER,OBJECT_VERSION_NUMBER,NAMESPACE_OBJ_ID,NAMESPACE_OBJ_VER.

- **PI_COMMENT** (Optional) Enter the reason you are checking in the Data Mart.

5.1.3 Modify a Data Mart

Use this API to modify a Data Mart definition or instance. You can modify the name and description. If you are modifying an instance object, you can also change the 3 REF attribute values to select a different source definition.

Note: If you are modifying a definition, you must first check it out.

**Name**  CDR_PUB_DF_DATAMART.ModifyDataMart

**Signature**

PROCEDURE MODIFYDATAMART(
  P_API_VERSION  IN    NUMBER,
  P_INIT_MSG_LIST  IN    VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_COMMIT  IN    VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_VALIDATION_LEVEL  IN    NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
  X_RETURN_STATUS  OUT    VARCHAR2,
  X_MSG_COUNT  OUT    NUMBER,
  X_MSG_DATA  OUT    VARCHAR2,
  PIO_CDRNAMING  IN OUT    CDR_NAMING_VERSION_OBJ_TYPE
);

**Parameters**  This API has standard parameters (see "Standard Parameters" on page 5) and the following parameters:

- **PIO_CDRNAMING** (Mandatory) This is a parameter of table type CDR_NAMING_VERSION_OBJ_TYPE that contains object attributes. Enter values to identify the Data Mart and enter new values for the attributes you want to modify. You can change the name or description for either a definition or instance. For an instance you can also
change to a different underlying source definition by entering values for the new
definition in the three REF attributes. All attributes are required.

NOTE: Use separate APIs for modifying the validation status and the version label:
CDR_PUB_VL_VALIDATION. UpdateValStatus and CDR_PUB_DF_NAMING. UpdateVersionLabel.

5.1.4 Check Out a Data Mart

Use this API to check out a Data Mart definition, either directly or through an instance
of it.

Name  CDR_PUB_DF_DATAMART.CheckOutDataMart

Signature

PROCEDURE CHECKOUTDATAMART(  
P_API_VERSION IN NUMBER,  
P_INIT_MSG_LIST IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,  
P_COMMIT IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,  
P_VALIDATION_LEVEL IN NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,  
X_RETURN_STATUS OUT VARCHAR2,  
X_MSG_COUNT OUT NUMBER,  
X_MSG_DATA OUT VARCHAR2,  
PIO_BASEOBJECT IN OUT CDR_BASE_OBJ_TYPE,  
PI_COMMENT IN VARCHAR2,  
PI_ISINSTONLY IN VARCHAR2  
);

Parameters  This API has standard parameters (see "Standard Parameters" on page 5)
and the following parameters:

■ PIO_BASEOBJECT (Mandatory) This is a parameter of table type CDR_BASE_  
OBJ_TYPE that contains object attributes. Enter values to identify the Data Mart  
that you want to do check out. I

f you are checking out the Data Mart definition directly, enter values to identify
the definition. If you are checking out a Data Mart definition through an instance
of it, enter values to identify the instance.

The following attributes are required: COMPANY_ID,OBJ_ID,OBJ_VER,OBJECT_  
VERSION_NUMBER,NAMESPACE_OBJ_ID,NAMESPACE_OBJ_VER.

The following attributes are required: COMPANY_ID, OBJ_ID, OBJ_VER. For  
OBJECT_TYPE_RC enter $OBJTYPES$WORKAREA. By default, new Work Areas  
receive a Usage Intent value of Development.

■ PI_COMMENT (Optional) Enter the reason you are checking out the Data Mart.

■ PI_ISINSTONLY (Mandatory) Enter $YESNO$NO.

5.1.5 Remove a Data Mart

Use this API to remove one or more Data Mart definitions or instances.

Name  CDR_PUB_DF_DATAMART.RemoveDataMart

Signature

PROCEDURE REMOVEDATAMART( 
P_API_VERSION IN NUMBER,
Parameters  This API has standard parameters (see "Standard Parameters" on page 5) and the following parameters:

PIO_CDRBASEOBJCOLL (Mandatory) This is a collection of CDR_BASE_OBJ_TYPEs. For each Data Mart that you want to remove, initialize a CDR_BASE_OBJ_TYPE and then extend the collection.

The following attributes are required: COMPANY_ID,OBJ_ID,OBJ_VER,OBJECT_VERSION_NUMBER,NAMESPACE_OBJ_ID,NAMESPACE_OBJ_VER.
This is a public interface for Domain-related operations, including creating, modifying, and removing Domains. It also includes functions for copying and moving objects into Domains.

6.1 Define and Modify Domains

This section contains the following topics:

- Section 6.1.1, "Create a Domain"
- Section 6.1.2, "Modify a Domain"
- Section 6.1.3, "Copy a Domain"
- Section 6.1.4, "Move Objects into a Domain"
- Section 6.1.5, "Copy Objects into a Domain"
- Section 6.1.6, "Copy Objects into a Domain and Check In"
- Section 6.1.7, "Remove a Domain"

6.1.1 Create a Domain

Use this API to create a new Domain.

**Name**  CDR_PUB_DF_DOMAIN.CreateDomain

**Signature**

```sql
PROCEDURE CREATEDOMAIN(
  P_API_VERSION IN NUMBER,
  P_INIT_MSG_LIST IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_COMMIT IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_VALIDATION_LEVEL IN NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
  X_RETURN_STATUS OUT VARCHAR2,
  X_MSG_COUNT OUT NUMBER,
  X_MSG_DATA OUT VARCHAR2,
  PIO_SOURCECDRNAMING IN OUT CDR_NAMING_VERSION_OBJ_TYPE,
  PO_DEFCLASSIFICATIONCOLL IN CDR_CLASSIFICATIONS_COLL
);
```

**Parameters**  This API has standard parameters (see "Standard Parameters" on page 5) and the following parameters:
- **PIO_SOURCECDRNAMING** (Mandatory) This is a parameter of table type CDR_NAMING_VERSIONS_OBJ_TYPE that contains CDR Naming Version attributes. For OBJECT_TYPE_RC enter $OBJTYPES$LIBDOMAIN.

- **PO_DEFCLASSIFICATIONCOLL** (Optional) By default the new Domain is classified according to the subtype you assigned it in the CDR_NAMING_VERSION_OBJ_TYPE.

  If you want to override the default classifications for one or more classification levels, use this parameter. This is a collection of CDR_CLA_OBJ_TYPEs, which have 5 attributes, including CLA_LEVEL_ID and CLASSIFICATION_ID. If you want the definition to inherit its classifications for a particular level from its parent, enter the classification level ID and, for the CLASSIFICATION_ID, enter 0 (zero).

  If you want to explicitly assign one or more terms for a particular level, initialize a CDR_CLA_OBJ_TYPE for each term, entering the classification level ID and, for the CLASSIFICATION_ID, the term ID. The PAR_ attributes are not relevant to Domains. Do not enter any values for them.

### 6.1.2 Modify a Domain

Use this API to modify a Domain.

**Name** CDR_PUB_DF_DOMAIN.ModifyDomain

**Signature**

```
PROCEDURE MODIFYDOMAIN(
    P_API_VERSION  IN    NUMBER,
    P_INIT_MSG_LIST  IN    VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_COMMIT  IN    VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_VALIDATION_LEVEL  IN    NUMBER := CDR_PUBL_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
    X_RETURN_STATUS  OUT    VARCHAR2,
    X_MSG_COUNT  OUT    NUMBER,
    X_MSG_DATA  OUT    VARCHAR2,
    PIO_SOURCECDRNAMING  IN OUT    CDR_NAMING_VERSION_OBJ_TYPE
);```

**Parameters** This API has standard parameters (see "Standard Parameters" on page 5) and the following parameters:

- **PIO_SOURCECDRNAMING** (Mandatory) This is a parameter of table type CDR_NAMING_VERSIONS_OBJ_TYPE that contains object attributes. Enter values to identify the Domain and enter new values for the attributes you want to modify. The COMPANY_ID, OBJ_ID and OBJ_VER of the domain to be modified should be initialized.

---

**Note:** Use separate APIs for modifying the validation status and the version label: CDR_PUB_VL_VALIDATION. UPDATEVALSTATUS and CDR_PUB_DF_NAMING. UPDATEVERSIONLABEL.

### 6.1.3 Copy a Domain

Use this API to make a copy of a Domain.

**Name** CDR_PUB_DF_DOMAIN.CopyDomain

---

6-2  Oracle Life Sciences Data Hub Application Programming Interface Guide
Define and Modify Domains

Signature

PROCEDURE COPYDOMAIN(
   P_API_VERSION IN NUMBER,
   P_INIT_MSG_LIST IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
   P_COMMIT IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
   P_VALIDATION_LEVEL IN NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
   X_RETURN_STATUS OUT VARCHAR2,
   X_MSG_COUNT OUT NUMBER,
   X_MSG_DATA OUT VARCHAR2,
   PI_CDRBASEOBJCOLL IN OUT CDR_BASE_OBJ_COLL
);

Parameters This API has standard parameters (see “Standard Parameters” on page 5) and the following parameters:

PI_CDRBASEOBJCOLL (Mandatory) This is a collection of CDR_BASE_OBJ_TYPES. For each Domain that you want to copy, initialize a CDR_BASE_OBJ_TYPE and then extend the collection.

The following attributes are required: COMPANY_ID,OBJ_ID,OBJ_VER,NAMESPACE_OBJ_ID,NAMESPACE_OBJ_VER,OBJECT_VERSION_NUMBER.

6.1.4 Move Objects into a Domain

Use this API to move subdomains, Application Areas, and object definitions from one Domain or Application Area, into another Domain.

Name CDR_PUB_DF_DOMAIN.MoveObjectsIntoDomain

Signature

PROCEDURE MOVEOBJECTSINTODOMAIN(
   P_API_VERSION IN NUMBER,
   P_INIT_MSG_LIST IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
   P_COMMIT IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
   P_VALIDATION_LEVEL IN NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
   X_RETURN_STATUS OUT VARCHAR2,
   X_MSG_COUNT OUT NUMBER,
   X_MSG_DATA OUT VARCHAR2,
   PI_CDRBASEOBJCOLL IN OUT CDR_BASE_OBJ_COLL,
   PI_CDRTARGETCONTAINEROBJECT IN OUT CDR_BASE_OBJ_TYPE
);

Parameters This API has standard parameters (see “Standard Parameters” on page 5) and the following parameters:

■ PI_CDRBASEOBJCOLL (Mandatory) This is a collection of CDR_BASE_OBJ_TYPES. For each object that you want to move, initialize a CDR_BASE_OBJ_TYPE and then extend the collection.

The following attributes are required: COMPANY_ID,OBJ_ID,OBJ_VER,OBJECT_VERSION_NUMBER,NAMESPACE_OBJ_ID,NAMESPACE_OBJ_VER,OBJECT_VERSION_NUMBER.

■ PI_CDRTARGETCONTAINEROBJECT (Mandatory) This is a parameter of CDR_BASE_OBJ_TYPE and contains basic naming details about the Domain into which you want move objects.

The following attributes are required: COMPANY_ID,OBJ_ID,OBJ_VER, NAMESPACE_OBJ_ID, NAMESPACE_OBJ_VER, and OBJECT_VERSION_NUMBER).
6.1.5 Copy Objects into a Domain

Use this API to copy objects into a Domain. You can copy subdomains, Application Areas, and object definitions from other Domains or Application Areas.

**Name**  
CDR_PUB_DF_DOMAIN.CopyObjectsIntoDomain

**Signature**

```sql
PROCEDURE COPYOBJECTSINTODOMAIN(
   P_API_VERSION  IN    NUMBER,
   P_INIT_MSG_LIST  IN    VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
   P_COMMIT  IN    VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
   P_VALIDATION_LEVEL  IN    NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
   X_RETURN_STATUS  OUT    VARCHAR2,
   X_MSG_COUNT  OUT    NUMBER,
   X_MSG_DATA  OUT    VARCHAR2,
   PI_CDRBASEOBJCOLL  IN OUT    CDR_BASE_OBJ_COLL,
   PI_CDRTARGETCONTAINEROBJECT  IN OUT    CDR_BASE_OBJ_TYPE
);```

**Parameters**  
This API has standard parameters (see "Standard Parameters" on page 5) and the following parameters:

- **PI_CDRBASEOBJCOLL** (Mandatory) This is a collection of CDR_BASE_OBJ_TYPES. For each object that you want to copy, initialize a CDR_BASE_OBJ_TYPE and then extend the collection.
  
The following attributes are required: COMPANY_ID,OBJ_ID,OBJ_VER,OBJECT_VERSION_NUMBER,NAMESPACE_OBJ_ID,NAMESPACE_OBJ_VER.

- **PI_CDRTARGETCONTAINEROBJECT** (Mandatory) This is a parameter of CDR_BASE_OBJ_TYPE and contains basic naming details about the Domain into which you want copy objects.
  
The following attributes are required: COMPANY_ID,OBJ_ID,OBJ_VER,NAMESPACE_OBJ_ID,NAMESPACE_OBJ_VER, and OBJECT_VERSION_NUMBER.

6.1.6 Copy Objects into a Domain and Check In

Use this overloaded API to check in the objects copied into a Domain (if they are checked out in the source Domain).

**Name**  
CDR_PUB_DF_DOMAIN.CopyObjectsIntoDomain

**Signature**

```sql
PROCEDURE COPYOBJECTSINTODOMAIN(
   P_API_VERSION  IN    NUMBER,
   P_INIT_MSG_LIST  IN    VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
   P_COMMIT  IN    VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
   P_VALIDATION_LEVEL  IN    NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
   X_RETURN_STATUS  OUT    VARCHAR2,
   X_MSG_COUNT  OUT    NUMBER,
   X_MSG_DATA  OUT    VARCHAR2,
   PI_CDRBASEOBJCOLL  IN OUT    CDR_BASE_OBJ_COLL,
   PI_CDRTARGETCONTAINEROBJECT  IN OUT    CDR_BASE_OBJ_TYPE,
   PI_CHECKINFLAG  IN    VARCHAR
);```
Parameters This API has standard parameters (see “Standard Parameters” on page 5) and the following parameters:

- **PI_CDRBASEOBJCOLL** (Mandatory) This is a collection of CDR_BASE_OBJ_TYPES. For each object that you want to copy, initialize a CDR_BASE_OBJ_TYPE and then extend the collection.

  The following attributes are required: COMPANY_ID, OBJ_ID, OBJ_VER, OBJECT_VERSION_NUMBER, NAMESPACE_OBJ_ID, NAMESPACE_OBJ_VER.

- **PI_CDRTARGETCONTAINEROBJECT** (Mandatory) This is a parameter of CDR_BASE_OBJ_TYPE and contains basic naming details about the Domain into which you want copy objects.

  The following attributes are required: COMPANY_ID, OBJ_ID, OBJ_VER, NAMESPACE_OBJ_ID, NAMESPACE_OBJ_VER, and OBJECT_VERSION_NUMBER.

- **PI_CHECKINFLAG** (Mandatory) Enter $YESNO$YES if you want to check in the copied objects, if they are checked out, and $YESNO$NO if you do not want to check in the copied objects.

### 6.1.7 Remove a Domain

Use this API to delete a Domain.

**Name** CDR_PUB_DF_DOMAIN.RemoveDomain

**Signature**

```sql
PROCEDURE REMOVEDOMAIN(
    P_API_VERSION IN NUMBER,
    P_INIT_MSG_LIST IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_COMMIT IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_VALIDATION_LEVEL IN NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
    X_RETURN_STATUS OUT VARCHAR2,
    X_MSG_COUNT OUT NUMBER,
    X_MSG_DATA OUT VARCHAR2,
    PIO_DOMAIN IN OUT CDR_BASE_OBJ_TYPE
);```

Parameters This API has standard parameters (see “Standard Parameters” on page 5) and the following parameters:

- **PIO_DOMAIN** (Mandatory) This is a parameter of CDR_BASE_OBJ_TYPE and contains basic naming details about the Domain you want to delete.

  The following attributes are required: COMPANY_ID, OBJ_ID, OBJ_VER, NAMESPACE_OBJ_ID, NAMESPACE_OBJ_VER, and OBJECT_VERSION_NUMBER.
This is a public interface for Load Set-related operations, including creating, modifying, and removing Load Sets. It also includes APIs for checking Load Sets in and out, undoing a Load Set checkout, and computing the status of a Load Set.

### 7.1 Define and Modify Load Sets

This section contains the following topics:

- Section 7.1.1, "Create a Load Set"
- Section 7.1.2, "Check Out a Load Set Definition"
- Section 7.1.3, "Modify a Load Set"
- Section 7.1.4, "Check In a Load Set Definition"
- Section 7.1.5, "Remove a Load Set"
- Section 7.1.6, "Synchronize Table Descriptors in a Load Set"

#### 7.1.1 Create a Load Set

Use this API to create a Load Set definition, Load Set instance or both.

**Name**  
CDR_PUB_DF_LOADSET.CreateLoadSet

**Signature**

```
PROCEDURE CREATELOADSET(
    P_API_VERSION IN NUMBER,
    P_INIT_MSG_LIST IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_COMMIT IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_VALIDATION_LEVEL IN NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
    X_RETURN_STATUS OUT VARCHAR2,
    X_MSG_COUNT OUT NUMBER,
    X_MSG_DATA OUT VARCHAR2,
    PIO_SOURCECDRNAMING IN OUT CDR_NAMING_VERSION_OBJ_TYPE,
    PI_CDRLOADSETOBJTYPE IN CDR_LOAD_SET_OBJ_TYPE,
    PI_CREATEOBJECT IN VARCHAR2,
    PI_INSTANCE_SUBTYPE_ID IN CDR_NAMINGS.OBJECT_SUBTYPE_ID%TYPE,
    PI_DEFCLASSIFICATIONCOLL IN CDR_CLASSIFICATIONS_COLL,
    PI_INSTCLASSIFICATIONCOLL IN CDR_CLASSIFICATIONS_COLL
);
```

**Parameters**  
This API has standard parameters (see "Standard Parameters" on page 5) and the following parameters:
Define and Modify Load Sets

- **PIO_SOURCECDRNAMING** *(Mandatory)* This is a parameter of table type CDR_NAMING_VERSIONS_OBJ_TYPE that contains CDR Naming Version attributes.

  If you are creating a new definition only or a new definition and an instance of it, enter values for the new definition. If you are creating an instance of an existing definition, enter values to identify the existing definition.

  For OBJECT_TYPE_RC enter $OBJTYPES$LOADSET if you are creating a definition only; $OBJTYPES$LOADSETREF if you are creating an instance of an existing definition; and NULL if you are creating a new definition and an instance of it.

- **PI_CDRLOADSETOBJTYPE** *(Mandatory)* This is a parameter of table type CDR_LOAD_SET_OBJ_TYPE that contains object attributes specific to Load Sets. Enter values for the Load Set that you want to create.

  The following attributes are required: COMPANY_ID, ADAPTER_COMPANY_ID, ADAPTER_OBJ_ID, ADAPTER_OBJ_VER.

- **PI_CREATEOBJECT** *(Mandatory)* Enter DEFN to create a definition only; INST to create an instance of an existing definition; or BOTH to create a definition and an instance of it.

- **PI_INSTANCE_SUBTYPE_ID** *(Optional)* If you are creating a new instance, enter the subtype ID that you want to give the instance.

  If you are creating a definition only, do not enter a value for this parameter.

- **PI_DEFCLASSIFICATIONCOLL** *(Optional)* By default, the new definition is classified according to the subtype you assigned it in the CDR_NAMING_VERSION_OBJ_TYPE.

  If you want to override the default classifications for one or more classification levels, use this parameter. This is a collection of CDR_CLA_OBJ_TYPES, which have 5 attributes, including CLA_LEVEL_ID and CLASSIFICATION_ID.

  If you want the definition to inherit its classifications for a particular level from its parent, enter the classification level ID and, for the CLASSIFICATION_ID, enter 0 (zero).

  If you want to explicitly assign one or more terms for a particular level, initialize a CDR_CLA_OBJ_TYPE for each term, entering the classification level ID and, for the CLASSIFICATION_ID, the term ID. The PAR_ attributes are not relevant to Load Sets. Do not enter any values for them.

  If you are not creating a new definition, do not enter values here.

- **PI_INSTCLASSIFICATIONCOLL** **PI_INSTANCE_SUBTYPE_ID**. If you want to override the default classifications for one or more classification levels, use this parameter. This is a collection of CDR_CLA_OBJ_TYPES, which have 5 attributes, including CLA_LEVEL_ID and CLASSIFICATION_ID.

  If you want the instance to inherit its classifications for a particular level from its parent Work Area, enter the classification level ID and, for the CLASSIFICATION_ID, enter 0 (zero).

  If you want to explicitly assign one or more terms for a particular level, initialize a CDR_CLA_OBJ_TYPE for each term, entering the classification level ID and, for the CLASSIFICATION_ID, the term ID. The PAR_ attributes are not relevant to Load Sets. Do not enter any values for them.

  If you are not creating a new instance, do not enter values here.
7.1.2 Check Out a Load Set Definition

Use this API to check out a Load Set definition.

**Name**  
CDR_PUB_DF_LOADSET.CheckOutLoadSet

**Signature**

```sql
PROCEDURE CHECKOUTLOADSET(
    P_API_VERSION  IN    NUMBER,
    P_INIT_MSG_LIST  IN    VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_COMMIT  IN    VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_VALIDATION_LEVEL  IN    NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
    X_RETURN_STATUS  OUT    VARCHAR2,
    X_MSG_COUNT  OUT    NUMBER,
    X_MSG_DATA  OUT    VARCHAR2,
    PIO_BASEOBJECT  IN OUT    CDR_BASE_OBJ_TYPE,
    PI_COMMENT  IN    VARCHAR2,
    PI_ISINSTONLY  IN    VARCHAR2
);
```

**Parameters**  
This API has standard parameters (see "Standard Parameters" on page 5) and the following parameters:

- **PIO_BASEOBJECT** *(Mandatory)* This is a parameter of table type CDR_BASE_OBJ_TYPE that contains object attributes. You can enter values to identify either the Load Set definition or an instance of it:
  - Pass the Load Set definition details if you want to check out and subsequently modify only the definition.
  - Pass the details of an instance of the Load Set definition if you want the instance to point to the new version of the Load Set definition.

  The following attributes are required: COMPANY_ID, OBJ_ID, OBJ_VER, OBJECT_VERSION_NUMBER, NAMESPACE_OBJ_ID, NAMESPACE_OBJ_VER.

  To get the OBJECT_VERSION_NUMBER, enter the following query: select Max(OBJECT_VERSION_NUMBER) from cdr_vl_val_docs_v where OBJ_ID = <objid> and OBJ_VER = <objver> and DOC_STATUS_RC = '$VALINFOSTATUS$ACTIVE';

  NAMESPACE_OBJ_ID. If you are entering information about the Load Set definition, enter the object ID of its containing Application Area. If you are entering information about the Load Set instance, enter the object ID of its containing Work Area.

  NAMESPACE_OBJ_VER. If you are entering information about the Load Set definition, enter the object version number of its containing Application Area. If you are entering information about the Load Set instance, enter the object version number of its containing Work Area.

- **PI_COMMENT** *(Optional)* Enter the reason you are checking out the Load Set.

- **PI_ISINSTONLY** Enter $YESNO$NO if you are checking out only the definition. Enter $YESNO$YES if you are checking out the definition through its instance.

7.1.3 Modify a Load Set

Use this API to modify a Load Set definition or instance. You can change the name, description, or version label.
Define and Modify Load Sets

---

**Note:** If you are modifying a definition, you must first check it out.

---

**Name**  CDR_PUB_DF_LOADSET.ModifyLoadSet

**Signature**

```sql
PROCEDURE MODIFYLOADSET(
    P_API_VERSION  IN    NUMBER,
    P_INIT_MSG_LIST  IN    VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_COMMIT  IN    VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_VALIDATION_LEVEL  IN    NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
    X_RETURN_STATUS  OUT    VARCHAR2,
    X_MSG_COUNT  OUT    NUMBER,
    X_MSG_DATA  OUT    VARCHAR2,
    PIO_CDRNAMING  IN OUT    CDR_NAMING_VERSION_OBJ_TYPE
);
```

**Parameters**  This API has standard parameters (see "Standard Parameters" on page 5) and the following parameters:

- **PIO_CDRNAMING** (Mandatory) This is a parameter of table type CDR_NAMING_VERSION_OBJ_TYPE that contains object attributes. Enter values to identify the Load Set and enter new values for the attributes you want to modify. All attributes are required.

---

**7.1.4 Check In a Load Set Definition**

Use this API to check in a Load Set definition.

**Name**  CDR_PUB_DF_LOADSET.CheckInLoadSetDefinition

**Signature**

```sql
PROCEDURE CHECKINLOADSETDEFINITION(
    P_API_VERSION  IN    NUMBER,
    P_INIT_MSG_LIST  IN    VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_COMMIT  IN    VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_VALIDATION_LEVEL  IN    NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
    X_RETURN_STATUS  OUT    VARCHAR2,
    X_MSG_COUNT  OUT    NUMBER,
    X_MSG_DATA  OUT    VARCHAR2,
    PIO_BASEOBJECT  IN OUT    CDR_BASE_OBJ_TYPE,
    PI_COMMENT  IN    VARCHAR2
);
```

**Parameters**  This API has standard parameters (see "Standard Parameters" on page 5) and the following parameters:

- **PIO_BASEOBJECT** (Mandatory) This is a parameter of table type CDR_BASE_OBJ_TYPE that contains object attributes. Enter values to identify the Load Set definition that you want to check in.

  The following attributes are required: COMPANY_ID,OBJ_ID,OBJ_VER,OBJECT_VERSION_NUMBER,NAMESPACE_OBJ_ID,NAMESPACE_OBJ_VER.

- **PI_COMMENT** (Optional) Enter the reason for checking in the Load Set.
7.1.5 Remove a Load Set

Use this API to remove one or more Load Set definitions or instances.

**Name**  
CDR_PUB_DF_LOADSET.RemoveLoadSet

**Signature**

```sql
PROCEDURE REMOVELOADSET(
    P_API_VERSION  IN    NUMBER,
    P_INIT_MSG_LIST  IN    VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_COMMIT  IN    VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_VALIDATION_LEVEL  IN    NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
    X_RETURN_STATUS  OUT    VARCHAR2,
    X_MSG_COUNT  OUT    NUMBER,
    X_MSG_DATA  OUT    VARCHAR2,
    PIO_CDRBASEOBJCOLL  IN OUT    CDR_BASE_OBJ_COLL
);```

**Parameters**  
This API has standard parameters (see "Standard Parameters" on page 5) and the following parameter:

- **PIO_CDRBASEOBJCOLL** *(Mandatory)* This is a collection of CDR_BASE_OBJ_TYPEs.

For each Load Set definition or instance that you want to remove, initialize a CDR_BASE_OBJ_TYPE and then extend the collection.

The following attributes are required: COMPANY_ID,OBJ_ID,OBJ_VER,OBJECT_VERSION_NUMBER,NAMESPACE_OBJ_ID,NAMESPACE_OBJ_VER.

7.1.6 Synchronize Table Descriptors in a Load Set

Use this API to synchronize all the existing table descriptors in a Loadset with the Table Definition at the source. When you use this API, it performs different tasks according to the situation:

- If the API finds no changes to be synchronized, it displays a message indicating that there are no changes to the table descriptors.
- If the API finds changes to be synchronized and the Load Set definition is checked out, it synchronizes the changes and displays a message indicating that changes were identified and synchronized.
- If the API finds changes to be synchronized and the Load Set definition is checked in, then it checks out the Load Set, completes synchronization and leaves the Load Set checked out.

Table descriptors (and corresponding Table definitions) are checked out only if changes are detected. After synchronization is complete, the API returns a collection of Table descriptor objIDs and obj versions that were changed.

The API does not create a new table descriptor in the Load Set but only synchronizes the existing Table Descriptors in the LS. The updated Table Descriptor generates the same outcome as when columns are uploaded for a Table Descriptor.

**Name**  
CDR_PUB_DF_LOADSET.SynchonizeTablesWithinLS

**Signature**

```sql
PROCEDURE SYCHRONIZETABLESWITHINLS(
    P_API_VERSION IN NUMBER,
    ```
Parameters This API has standard parameters (see "Standard Parameters" on page 5) and the following parameter:

- **PARAM P_API_VERSION (Mandatory).** Enter the current version of the API you are calling. The API compares the version numbers of incoming calls to its current version number and returns an error if they are incompatible.

- **PARAM P_INIT_MSG_LIST (Optional).** Accept the default value (FND_API.G_FALSE) to ensure that this individual API does not initialize the message list upon completion. Pass FND_API.G_TRUE to override the default behavior.

- **PARAM P_COMMIT (Optional).** Accept the default value (FND_API.G_FALSE) to ensure that this individual API does not commit upon completion. Pass FND_API.G_TRUE to override the default behavior.

- **PARAM P_VALIDATION_LEVEL (Optional).** Accept the default value to perform full validation. No other values are currently supported.

- **PARAM X_RETURN_STATUS.** This output parameter returns the end status of the API: (S) Success, (E) Error or (U) Unexpected Error

- **PARAM X_MSG_COUNT.** This output parameter returns the count of error messages if the return status is other than Success.

- **PARAM X_MSG_DATA.** If the message count is 1, this output parameter returns the text of the error message. If there are more than one message, use cdr_pub_msg_pub.get to retrieve the messages.

- **PARAM PI_NLSDEFCOMPID.** Enter the Company Id.

- **PARAM PI_NLSDEFID.** Provide the Load Set Definition Id.

- **PARAM PI_VFILENAME (Optional).** If the Load Set is a SAS or TEXT type, provide the file name.
  
  For SAS—cport. xport, cprt, xprt and .sas*dat files are allowed.
  
  For TXT—zip and mdd files are allowed. The zip should only contain mdd files. The mdd file name should be the same as the Table Descriptor it needs to update. For other types, it can be sent as null.

- **PARAM PI_BFILEBLOB (Optional).** Provide the BLOB that holds the file content for SAS and TEXT Load Sets.

- **PARAM PO_TDESCLIST.** This list contains the table descriptors that were updated.

- **PARAM PO_OUTMSG.** Returns the error/confirmation messages.
This is a public interface for operations involving Parameter sets. These APIs provide procedures to create Parameter sets, modify Parameter set details, remove Parameter sets, check in, check out and uncheckout Parameter sets.

8.1 Create and Modify Parameter Sets

This section contains the following topics:

- Section 8.1.1, "Create a Parameter Set Definition"
- Section 8.1.2, "Check Out a Parameter Set Definition"
- Section 8.1.3, "Modify a Parameter Set Definition"
- Section 8.1.4, "Modify a Parameter Set Detail"
- Section 8.1.5, "Check In a Parameter Set Definition"
- Section 8.1.6, "Remove a Parameter Set Definition"

8.1.1 Create a Parameter Set Definition

Use this API to create a new Parameter Set definition, a new instance of an existing Parameter Set definition, or a new definition and an instance of it.

**Name**  CDR_PUB_DF_PARAMETER_SET.CreateParameterSet

**Signature**

PROCEDURE CREATEPARAMETERSET(  
  P_API_VERSION IN NUMBER,  
  P_INIT_MSG_LIST IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,  
  P_COMMIT IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,  
  P_VALIDATION_LEVEL IN NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,  
  X_RETURN_STATUS OUT VARCHAR2,  
  X_MSG_COUNT OUT NUMBER,  
  X_MSG_DATA OUT VARCHAR2,  
  PIO_SOURCECDRNAMING IN OUT CDR_NAMING_VERSION_OBJ_TYPE,  
  PI_CDRPSOBJECTYPE IN CDR_PARAM_SETS_OBJ_TYPE,  
  PI_CREATEOBJECT IN VARCHAR2,  
  PI_INSTANCE_SUBTYPE_ID IN CDR_NAMINGS.OBJECT_SUBTYPE_ID%TYPE,  
  PI_DEFCLASSIFICATIONCOLL IN CDR_CLASSIFICATIONS_COLL 
);

**Parameters**  This API has standard parameters (see "Standard Parameters" on page 5) and the following parameters:
Create and Modify Parameter Sets

- **PIO_SOURCECDRNAME** (Mandatory) This is a parameter of table type CDR_NAMING_VERSIONS_OBJ_TYPE that contains CDR Naming Version attributes.

  If you are creating a new definition only or a new definition and an instance of it, enter values for the new definition.

  If you are creating an instance of an existing definition, enter values to identify the definition. For OBJECT_TYPE_RC enter $OBJTYPES$PARAMSET if you are creating a definition only; $OBJTYPES$PARAMSETREF if you are creating an instance of an existing definition; and NULL if you are creating a new definition and an instance of it.

- **PI_CDRPSOBJTYPE** (Optional) This is a parameter of table type CDR_PARAM_SETS_OBJ_TYPE that contains object attributes specific to Parameter Sets.

  If you are creating a new definition, enter values for the new Parameter Set.

  The following attributes are required: COMPANY_ID, USAGE, PR_REF_ID, PR_REF_VER.

  If you are creating an instance of an existing Parameter Set, do not enter any values here.

- **PI_CREATEOBJECT** (Mandatory) Enter DEFN to create a definition only; INST to create a instance of an existing definition; or BOTH to create a new definition and an instance of it.

- **PI_INSTANCE_SUBTYPE_ID** (Optional) If you are creating a new instance, enter the ID for the subtype you want to give the instance.

  If you are creating a definition only, do not enter a value for this parameter.

- **PI_DEFCLASSIFICATIONCOLL** (Optional) By default, the new definition is classified according to the subtype you assigned it in the CDR_NAMING_VERSIONS_OBJ_TYPE.

  If you want to override the default classifications for one or more classification levels, use this parameter. This is a collection of CDR_CLA_OBJ_TYPES, which have 5 attributes, including CLA_LEVEL_ID and CLASSIFICATION_ID.

  If you want the definition to inherit its classifications for a particular level from its parent, enter the classification level ID and, for the CLASSIFICATION_ID, enter 0 (zero).

  If you want to explicitly assign one or more terms for a particular level, initialize a CDR_CLA_OBJ_TYPE for each term, entering the classification level ID and, for the CLASSIFICATION_ID, the term ID. The PAR attributes are not relevant to Parameter Sets. Do not enter any values for them.

  If you are not creating a new definition, do not enter values here.

### 8.1.2 Check Out a Parameter Set Definition

Use this API to check out a Parameter Set definition either directly or through an instance of it.

**Name**  
CDR_PUB_DF_PARAMETER_SET.CheckOutParameterSet

**Signature**

```sql
PROCEDURE CHECKOUTPARAMETERSET(
P_API_VERSION IN NUMBER,
```
Create and Modify Parameter Sets

8.3 Modify a Parameter Set Definition

Use this API to modify the name or description of a Parameter Set definition.

**Name**  
CDR_PUB_DF_PARAMETER_SET.ModifyParameterSetDefinition

**Signature**

PROCEDURE MODIFYPARAMETERSETDEFINITION(  
P_API_VERSION IN NUMBER,  
P_INIT_MSG_LIST IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,  
P_COMMIT IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,  
P_VALIDATION_LEVEL IN NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,  
X_RETURN_STATUS OUT VARCHAR2,  
X_MSG_COUNT OUT NUMBER,  
X_MSG_DATA OUT VARCHAR2,  
PIO_CDRNAMING IN OUT CDR_NAMING_VERSION_OBJ_TYPE  
);

**Parameters**  
This API has standard parameters (see “Standard Parameters” on page 5) and the following parameter:

- **PIO_CDRNAMING** (Mandatory) This is a parameter of table type CDR_NAMING_VERSION_OBJ_TYPE that contains object attributes. Enter values to identify the Parameter Set and enter new values for the attributes you want to modify. You can change the name or description.
8.1.4 Modify a Parameter Set Detail

Use this API to modify Parameter Set details.

**Name**  CDR_PUB_DF_PARAMETER_SET.ModifyParameterSetDetails

**Signature**

```sql
PROCEDURE MODIFYPARAMETERSETDETAILS(
    P_API_VERSION  IN    NUMBER,
    P_INIT_MSG_LIST  IN    VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_COMMIT  IN    VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_VALIDATION_LEVEL  IN    NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
    X_RETURN_STATUS  OUT    VARCHAR2,
    X_MSG_COUNT  OUT    NUMBER,
    X_MSG_DATA  OUT    VARCHAR2,
    PIO_CDRNAMING  IN OUT    CDR_NAMING_VERSION_OBJ_TYPE
);
```

**Parameters**  This API has standard parameters (see "Standard Parameters" on page 5) and the following parameter:

- **PIO_CDRNAMING**  Naming details of the parameter set object to be modified.

8.1.5 Check In a Parameter Set Definition

Use this API to check in a Parameter Set definition.

**Name**  CDR_PUB_DF_PARAMETER_SET.CheckInParameterSetDefinition

**Signature**

```sql
PROCEDURE CHECKINPARAMETERSETDEFINITION(
    P_API_VERSION  IN    NUMBER,
    P_INIT_MSG_LIST  IN    VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_COMMIT  IN    VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_VALIDATION_LEVEL  IN    NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
    X_RETURN_STATUS  OUT    VARCHAR2,
    X_MSG_COUNT  OUT    NUMBER,
    X_MSG_DATA  OUT    VARCHAR2,
    PIO_BASEOBJECT  IN OUT    CDR_BASE_OBJ_TYPE,
    PI_COMMENT  IN    VARCHAR2
);
```

**Parameters**  This API has standard parameters (see "Standard Parameters" on page 5) and the following parameters:

- **PIO_BASEOBJECT**  (Mandatory)  This is a parameter of table type CDR_BASE_OBJ_TYPE that contains object attributes. Enter values to identify the Parameter Set definition that you want to check in.

  The following attributes are required: COMPANY_ID,OBJ_ID,OBJ_VER,OBJECT_VERSION_NUMBER,NAMESPACE_OBJ_ID,NAMESPACE_OBJ_VER.
PI_COMMENT (Optional) Enter the reason you are checking in the Parameter Set.

8.1.6 Remove a Parameter Set Definition

Use this API to remove one or more Parameter Set definitions or instances.

**Name**  CDR_PUB_DF_PARAMETER_SET.RemoveParameterSet

**Signature**

PROCEDURE REMOVEPARAMETERSET(
  P_API_VERSION  IN    NUMBER,
  P_INIT_MSG_LIST  IN    VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_COMMIT  IN    VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_VALIDATION_LEVEL  IN    NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
  X_RETURN_STATUS  OUT    VARCHAR2,
  X_MSG_COUNT  OUT    NUMBER,
  X_MSG_DATA  OUT    VARCHAR2,
  PIO_CDRBASEOBJCOLL  IN OUT    CDR_BASE_OBJ_COLL
);

**Parameters**  This API has standard parameters (see "Standard Parameters" on page 5) and the following parameter:

PIO_CDRBASEOBJCOLL (Mandatory) This is a collection of CDR_BASE_OBJ_TYPES.

For each Parameter Set that you want to remove, initialize a CDR_BASE_OBJ_TYPE and then extend the collection.

The following attributes are required: COMPANY_ID,OBJ_ID,OBJ_VER,OBJECT_VERSION_NUMBER,NAMESPACE_OBJ_ID,NAMESPACE_OBJ_VER.
This is a public interface for all Planned Output related functions including creating, modifying, removing, and copying Planned Outputs.

9.1 Create and Modify Planned Outputs

This section contains the following topics:

- Section 9.1.1, "Create a Planned Output"
- Section 9.1.2, "Get a New Position Number"
- Section 9.1.3, "Get a Planned Output Object"
- Section 9.1.4, "Modify a Planned Output"
- Section 9.1.5, "Identify whether a SAS Object"
- Section 9.1.6, "Remove a Planned Output Object"

9.1.1 Create a Planned Output

Use this API to create a Planned Output.

**Name**  CDR_PUB_DF_PLANNED_OUTPUT.CreatePlannedOutput

**Signature**

PROCEDURE CREATEPLANNEDOUTPUT(
  P_API_VERSION IN    NUMBER,
  P_INIT_MSG_LIST IN    VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_COMMIT IN    VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_VALIDATION_LEVEL IN    NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
  X_RETURN_STATUS  OUT    VARCHAR2,
  X_MSG_COUNT  OUT    NUMBER,
  X_MSG_DATA  OUT    VARCHAR2,
  PIO_SOURCECDRNAMING IN OUT    CDR_NAMING_VERSIONS_OBJ_TYPE,
  PI_PLANNEDOUTPUTOBJTYPE IN OUT    CDR_PLANNEDOUTPUT_OBJ_TYPE,
  PI_DEFCLASSIFICATIONCOLL IN    CDR_CLASSIFICATIONS_COLL
);

**Parameters**  This API has standard parameters (see "Standard Parameters" on page 5) and the following parameters:

- **PIO_SOURCECDRNAMING** (Mandatory) This is a parameter of type CDR_NAMING_VERSIONS_OBJ_TYPE.
Create and Modify Planned Outputs

The following attributes are required: COMPANY_ID, OBJECT_TYPE_RC, NAME, NAMESPACE_OBJ_ID, NAMESPACE_OBJ_VER, OWNING_LOCATION_RC, OBJECT_SUBTYPE_ID, DESCRIPTION, REF_COMPANY_ID, REF_OBJ_ID, REF_OBJ_VER.

For OBJECT_TYPE_RC enter $OBJTYPES$PLANNEDOUT.

- **PI_PLANNEDOUTPUTOBJTYPE** (Mandatory) This is a parameter of table type CDR_PLANNEDOUTPUT_OBJ_TYPE.

  You need to pass the following attributes: COMPANY_ID, OBJ_ID, OBJ_VER, TITLE, POSITION, FILEREF, FILE_NAME, PRIMARY_FLAG_RC, ERROR_FLAG_RC, REQ_FLAG_RC.

  For the POSITION attribute, use the GETNEWPOSITIONNUMBER API.

- **PI_DEFCLASSIFICATIONCOLL** (Optional) By default the new PLANNED OUTPUT is classified according to the subtype you assigned it in the CDR_NAMING_VERSION_OBJ_TYPE.

  If you want to override the default classifications for one or more classification levels, use this parameter. This is a collection of CDR_CLA_OBJ_TYPES, which have 5 attributes, including CLA_LEVEL_ID and CLASSIFICATION_ID.

  If you want the definition to inherit its classifications for a particular level from its parent, enter the classification level ID and, for the CLASSIFICATION_ID, enter 0 (zero).

  If you want to explicitly assign one or more terms for a particular level, initialize a CDR_CLA_OBJ_TYPE for each term, entering the classification level ID and, for the CLASSIFICATION_ID, the term ID.

  If you want to use a Parameter to classify the actual output, use the PAR attributes to identify the Parameter. The Parameter’s list of values must be based on a classification hierarchy level.

### 9.1.2 Get a New Position Number

Use this API to get a unique position number for a Planned Output within a parent object. This API is used from within the CREATEPLANNEDOUTPUT API, which is used to create a new Planned Output. You can also reorder Planned Outputs using this API.

**Name**  
CDR_PUB_DF_PLANNED_OUTPUT.GetNewPositionNumber

**Signature**

FUNCTION GETNEWPOSITIONNUMBER(
   P_API_VERSION  IN    NUMBER,
   P_INIT_MSG_LIST  IN    VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
   P_COMMIT  IN    VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
   P_VALIDATION_LEVEL  IN    NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
   PI_NCOMPANYID  IN    CDR_NAMINGS.COMPANY_ID%TYPE,
   PI_NOBJID  IN    CDR_NAMINGS.OBJ_ID%TYPE,
   PI_NOBJVER  IN    CDR_NAMING_VERSIONS.OBJ_VER%TYPE
) RETURN NUMBER;

**Parameters**  
This API has standard parameters (see "Standard Parameters" on page 5) and the following parameters:

- **PI_NCOMPANYID** (Mandatory) Enter the COMPANY_ID of the parent object.
Create and Modify Planned Outputs

- **PI_NOBJID** (Mandatory) Enter the OBJ_ID of the parent object.
- **PI_NOBJVER** (Mandatory) Enter the OBJ_VER of the parent object.

### 9.1.3 Get a Planned Output Object

Use this API to retrieve a Planned Output object for an LSH object.

**Name**  CDR_PUB_DF_PLANNED_OUTPUT.GetPlannedOutputObject

**Signature**

```sql
FUNCTION GETPLANNEDOUTPUTOBJECT(  
  P_API_VERSION IN NUMBER,  
  P_INIT_MSG_LIST IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,  
  P_COMMIT IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,  
  P_VALIDATION_LEVEL IN NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,  
  PI_NCOMPANYID IN CDR_NAMINGS.COMPANY_ID%TYPE,  
  PI_NOBJID IN CDR_NAMINGS.OBJ_ID%TYPE,  
  PI_NOBJVER IN CDR_NAMING_VERSIONS.OBJ_VER%TYPE,  
  PO_PLANNEDOUTPUT OUT CDR_PLANNEDOUTPUT_OBJ_TYPE  
) RETURN BOOLEAN;
```

**Return Type** BOOLEAN

**Description**

**Parameters** This API has standard parameters (see "Standard Parameters" on page 5) and the following parameters:

- **PI_NCOMPANYID** (Mandatory) Enter your company ID. To get your company ID, use CDR_DF_PUB_DEF_CONSTANTS.CURRENT_COMPANY_ID.
- **PI_NOBJID** (Mandatory) Enter the OBJ_ID value of the object for which you want to retrieve the Planned Output object.
- **PI_NOBJVER** (Mandatory) Enter the OBJ_VER value of the object for which you want to retrieve the Planned Output's object.
- **PO_PLANNEDOUTPUT** This is the output parameter of the API returning the Planned Output Object.

### 9.1.4 Modify a Planned Output

Use this API to modify an existing Planned Output.

**Name**  CDR_PUB_DF_PLANNED_OUTPUT.ModifyPlannedOutput

**Signature**

```sql
PROCEDURE MODIFYPLANNEDOUTPUT(  
  P_API_VERSION IN NUMBER,  
  P_INIT_MSG_LIST IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,  
  P_COMMIT IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,  
  P_VALIDATION_LEVEL IN NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,  
  X_RETURN_STATUS OUT VARCHAR2,  
  X_MSG_COUNT OUT NUMBER,  
  X_MSG_DATA OUT VARCHAR2,  
  PI_SOURCECDRNAMING IN OUT CDR_NAMING_VERSION_OBJ_TYPE,  
  PI_POUTOBJTYPE IN OUT CDR_PLANNEDOUTPUT_OBJ_TYPE
  ) ;
```

Planned Outputs 9-3
Parameters  This API has standard parameters (see "Standard Parameters" on page 5) and the following parameters:

- **PI_SOURCECDRNAMING** (Mandatory) This is a parameter of table type CDR_NAMING_VERSION_OBJ_TYPE that contains object attributes. Enter values to identify the Planned Output.

- **PI_PUTOBJTYPE** (Mandatory) This is a parameter of table CDR_PLANNEDOUTPUT_OBJ_TYPE that contains attributes specific to Planned Outputs. Enter values for the Planned Output and enter new values for the attributes you want to modify. You can change the name or description of a Planned Output.

---

Note: Use separate APIs for modifying the validation status and the version label: CDR_PUB_VL_VALIDATION.UPDATE_VAL_STATUS and CDR_PUB_DF_NAMING.UPDATEVERSIONLABEL.

---

9.1.5 Identify whether a SAS Object

Use this API to check if the parent object of the Planned Output is a SAS object. SAS objects are: SAS programs, SAS load Sets, SAS Tables.

**Name**  CDR_PUB_DF_PLANNED_OUTPUT.IsSASObject

**Signature**

FUNCTION ISSASOBJECT(
  P_API_VERSION IN NUMBER,
  P_INIT_MSG_LIST IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_COMMIT IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_VALIDATION_LEVEL IN NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
  PI_NCOMPANYID IN CDR_NAMINGS.COMPANY_ID%TYPE,
  PI_NOBJID IN CDR_NAMINGS.OBJ_ID%TYPE,
  PI_NOBJVER IN CDR_NAMING_VERSIONS.OBJ_VER%TYPE,
  PI_NSOBJTYPE IN CDR_NAMINGS.OBJECT_TYPE_RC%TYPE
) RETURN VARCHAR2;

**Return**  Type VARCHAR2

**Description**

Parameters  This API has standard parameters (see "Standard Parameters" on page 5) and the following parameters:

- **PI_NCOMPANYID** (Mandatory) Enter your company ID.

  To get your company ID, use CDR_PUB_DEF_FACTORY_UTILS.GetCompanyId.

- **PI_NOBJID** (Mandatory) Enter the object ID of the Planned Output’s parent object.

- **PI_NOBJVER** (Mandatory) Enter the version number of the Planned Output’s parent object.

- **PI_NSOBJTYPE** (Mandatory) Enter the OBJ_TYPE value for the Planned Output’s parent object.

---

9.1.6 Remove a Planned Output Object

Use this API to delete one or more Planned Outputs.
Name  CDR_PUB_DF_PLANNED_OUTPUT.RemovePlannedOutput

Signature

PROCEDURE REMOVEPLANNEDOUTPUT(
    P_API_VERSION IN NUMBER,
    P_INIT_MSG_LIST IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_COMMIT IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_VALIDATION_LEVEL IN NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
    X_RETURN_STATUS OUT VARCHAR2,
    X_MSG_COUNT OUT NUMBER,
    X_MSG_DATA OUT VARCHAR2,
    PIO_CDRBASEOBJCOLL IN OUT CDR_BASE_OBJ_COLL,
    PI_GETLOCK IN VARCHAR := 'T'
);

Parameters  This API has standard parameters (see "Standard Parameters" on page 5) and the following parameters:

- **PIO_CDRBASEOBJCOLL** (Mandatory) This is a collection of CDR_BASE_OBJ_TYPES.

  For each Planned Output that you want to remove, initialize a CDR_BASE_OBJ_TYPE and then extend the collection.

  The following attributes are required: COMPANY_ID,OBJ_ID,OBJ_VER,OBJECT_VERSION_NUMBER,NAMESPACE_OBJ_ID,NAMESPACE_OBJ_VER.

- **PI_GETLOCK** (Optional) Do not enter a value for this attribute. The default and only acceptable value is 'T'.
This is a public interface for Program-related functions, including creating, modifying, and removing Programs. It also includes functions for checking Programs in and out, and undoing a Program checkout.

10.1 Create and Modify Programs

This section contains the following topics:

- Section 10.1.1, "Create a Program"
- Section 10.1.2, "Copy Objects Into a Program"
- Section 10.1.3, "Modify a Program"
- Section 10.1.4, "Check In a Program Definition"
- Section 10.1.5, "Check Out a Program Definition"
- Section 10.1.6, "Remove a Program"
- Section 10.1.7, "Create a Planned Output for a Log File"
- Section 10.1.8, "Assign a Planned Output"
- Section 10.1.9, "Modify a Manual Validation Flag Value"

10.1.1 Create a Program

Use this API to create a new Program definition, a new instance of an existing Program definition, or a new definition and an instance of it.

**Name**  CDR_PUB_DF_PROGRAM.CreateProgram

**Signature**

```sql
PROCEDURE CREATEPROGRAM(
    P_API_VERSION IN    NUMBER,
    P_INIT_MSG_LIST IN    VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_COMMIT IN    VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_VALIDATION_LEVEL IN    NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
    X_RETURN_STATUS OUT    VARCHAR2,
    X_MSG_COUNT OUT    NUMBER,
    X_MSG_DATA OUT    VARCHAR2,
    PIO_SOURCECDRNAMING IN OUT    CDR_NAMING_VERSION_OBJ_TYPE,
    PI_CDRPRGOBJTYPE IN    CDR_PROGRAM_OBJ_TYPE,
    PI_CREATEOBJECT IN    VARCHAR2,
    PI_INSTANCE_SUBTYPE_ID IN    CDR_NAMINGS.OBJECT_SUBTYPE_ID%TYPE,
) RETURN;
```
Parameters  This API has standard parameters (see "Standard Parameters" on page 5) and the following parameters:

- **PI_SOURCECDRNAME** (Mandatory) This is a parameter of table type CDR_NAMING_VERSIONS_OBJ_TYPE that contains CDR Naming Version attributes.
  
  If you are creating a new definition only or a new definition and an instance of it, enter values for the new definition.
  
  If you are creating an instance of an existing definition, enter values to identify the definition.
  
  The required attributes are: OBJECT_TYPE_RC, NAME, NAMESPACE_OBJ_ID, NAMESPACE_OBJ_VER, NAMESPACE_START_OBJ_VER, NAMESPACE_END_OBJ_VER, OWNING_LOCATION_RC, OBJECT_SUBTYPE_ID.
  
  For OBJECT_TYPE_RC enter $OBJTYPES$PROGRAM if you are creating a definition only; $OBJTYPES$PROGRAMINST if you are creating an instance of an existing definition; and NULL if you are creating a new definition and an instance of it.

- **PI_CDRPRGOBJTYPE** (Optional) This is a parameter of table type CDR_DATA_MART_OBJ_TYPE that contains object attributes specific to Programs.
  
  If you are creating a new definition, enter values for the new Program. The following attributes are required: CDR_PROGRAM_OBJ_TYPE, TECH_TYPE_ID.
  
  If you are creating an instance of an existing Program, do not enter any values here.

- **PI_CREATEOBJECT** (Mandatory) Enter DEFN to create a definition only; INST to create an instance of an existing definition; or BOTH to create a new definition and an instance of it.

- **PI_INSTANCE_SUBTYPE_ID** (Optional) If you are creating a new instance, enter the ID for the subtype you want to give the instance.
  
  If you are creating a definition only, do not enter a value for this parameter.
  
  Definition and Instance are to be created in the same call.

- **PI_DEFCLASSIFICATIONCOLL** (Optional) By default the new definition is classified according to the subtype you assigned it in the CDR_NAMING_VERSION_OBJ_TYPE.
  
  If you want to override the default classifications for one or more classification levels, use this parameter. This is a collection of CDR_CLA_OBJ_TYPES, which have five attributes, including CLA_LEVEL_ID and CLASSIFICATION_ID.
  
  If you want the definition to inherit its classifications for a particular level from its parent, enter the classification level ID and, for the CLASSIFICATION_ID, enter 0 (zero).
  
  If you want to explicitly assign one or more terms for a particular level, initialize a CDR_CLA_OBJ_TYPE for each term, entering the classification level ID and, for the CLASSIFICATION_ID, the term ID. Do not enter any values for them.
  
  If you are not creating a new definition, do not enter values here.


- **PI_INSTCLASSIFICATIONCOLL** (Optional) By default the new instance is classified according to the subtype you assigned it in the CDR_NAMING_VERSION_OBJ_TYPE.

If you want to override the default classifications for one or more classification levels, use this parameter. This is a collection of CDR_CLA_OBJ_TYPES, which have 5 attributes, including CLA_LEVEL_ID and CLASSIFICATION_ID.

If you want the instance to inherit its classifications for a particular level from its parent Work Area, enter the classification level ID and, for the CLASSIFICATION_ID, enter 0 (zero).

If you want to explicitly assign one or more terms for a particular level, initialize a CDR_CLA_OBJ_TYPE for each term, entering the classification level ID and, for the CLASSIFICATION_ID, the term ID. The PAR_ attributes are not relevant to Programs. Do not enter any values for them.

If you are not creating a new instance, do not enter values here.

### 10.1.2 Copy Objects Into a Program

Use this API to copy objects into a Program.

**Name**  CDR_PUB_DF_PROGRAM.CopyObjectIntoProgram

**Signature**

```
PROCEDURE COPYOBJECTINTOPROGRAM(
  P_API_VERSION  IN    NUMBER,
  P_INIT_MSG_LIST  IN    VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_COMMIT  IN    VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_VALIDATION_LEVEL  IN    NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
  X_RETURN_STATUS  OUT    VARCHAR2,
  X_MSG_COUNT  OUT    NUMBER,
  X_MSG_DATA  OUT    VARCHAR2,
  PI_CDRBASEOBJCOLL  IN    CDR_BASE_OBJ_COLL,
  PI_CDRTARGETCONTAINEROBJECT  IN OUT    CDR_BASE_OBJ_TYPE,
  PI_CHECKINFLAG  IN    VARCHAR2);
```

**Parameters**  This API has standard parameters (see "Standard Parameters" on page 5) and the following parameters:

- **PI_CDRBASEOBJCOLL** Collection of objects to be copied.
- **PI_CDRTARGETCONTAINEROBJECT** The target container object.
- **PI_CHECKINFLAG** Flag to indicate if the copied objects are to be checked in.

### 10.1.3 Modify a Program

Use this API to modify a Program definition or instance. You can modify the name and description.

If you are modifying an instance object, you can also change the 3 REF attribute values to select a different source definition.

**Note:**  Note: If you are modifying a definition, you must first check it out.
Name  CDR_PUB_DF_PROGRAM.ModifyProgramDetails

Signature

PROCEDURE MODIFYPROGRAMDETAILS(
   P_API_VERSION  IN    NUMBER,
   P_INIT_MSG_LIST  IN    VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
   P_COMMIT  IN    VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
   P_VALIDATION_LEVEL  IN    NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
   X_RETURN_STATUS  OUT    VARCHAR2,
   X_MSG_COUNT  OUT    NUMBER,
   X_MSG_DATA  OUT    VARCHAR2,
   PIO_CDRNAMING  IN OUT    CDR_NAMING_VERSION_OBJ_TYPE
);

Parameters  This API has standard parameters (see "Standard Parameters" on page 5) and the following parameter:

PIO_CDRNAMING (Mandatory) This is an IN OUT parameter of table type CDR_NAMING_VERSIONS_OBJ_TYPE that contains CDR Naming Version attributes.

10.1.4 Check In a Program Definition

Use this API to check in a Program definition.

Name  CDR_PUB_DF_PROGRAM.CheckInProgramDefinition

Signature

PROCEDURE CHECKINPROGRAMDEFINITION(
   P_API_VERSION  IN    NUMBER,
   P_INIT_MSG_LIST  IN    VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
   P_COMMIT  IN    VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
   P_VALIDATION_LEVEL  IN    NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
   X_RETURN_STATUS  OUT    VARCHAR2,
   X_MSG_COUNT  OUT    NUMBER,
   X_MSG_DATA  OUT    VARCHAR2,
   PIO_BASEOBJECT  IN OUT    CDR_BASE_OBJ_TYPE,
   PI_COMMENT  IN    VARCHAR2
);

Parameters  This API has standard parameters (see "Standard Parameters" on page 5) and the following parameters:

■ PIO_BASEOBJECT (Mandatory) This is a parameter of table type CDR_BASE_OBJ_TYPE that contains object attributes. Enter values to identify the Program definition that you want to check in.

   The following attributes are required: COMPANY_ID,OBJ_ID,OBJ_VER,OBJECT_VERSION_NUMBER,NAMESPACE_OBJ_ID,NAMESPACE_OBJ_VER.

■ PI_COMMENT (Optional) Enter the reason you are checking in the Program.

10.1.5 Check Out a Program Definition

Use this API to check out a Program definition either directly or through an instance of it.

Name  CDR_PUB_DF_PROGRAM.CheckOutProgram
Create and Modify Programs

**Signature**

PROCEDURE CHECKOUTPROGRAM(
    P_API_VERSION IN NUMBER,
    P_INIT_MSG_LIST IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_COMMIT IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_VALIDATION_LEVEL IN NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
    X_RETURN_STATUS OUT VARCHAR2,
    X_MSG_COUNT OUT NUMBER,
    X_MSG_DATA OUT VARCHAR2,
    PIO_BASEOBJECT IN OUT CDR_BASE_OBJ_TYPE,
    PI_COMMENT IN VARCHAR2,
    PI_ISINSTONLY IN VARCHAR2
);

**Parameters**  This API has standard parameters (see “Standard Parameters” on page 5) and the following parameters:

- **PIO_BASEOBJECT** (Mandatory) This is a parameter of table type CDR_BASE_OBJ_TYPE that contains object attributes. Enter values to identify the Program that you want to check out.
  
  If you are checking out the Program definition directly, enter values to identify the definition.
  
  If you are checking out a Program definition through an instance of it, enter values to identify the instance.
  
  The following attributes are required: COMPANY_ID,OBJ_ID,OBJ_VER,OBJECT_VERSION_NUMBER,NAMESPACE_OBJ_ID,NAMESPACE_OBJ_VER.

- **PI_COMMENT** (Optional) Enter the reason you are checking out the Program.

- **PI_ISINSTONLY** Enter $YESNO$NO.

**10.1.6 Remove a Program**

Use this API to remove one or more Program definitions or instances.

**Name**  CDR_PUB_DF_PROGRAM.RemoveProgram

**Signature**

PROCEDURE REMOVEPROGRAM(
    P_API_VERSION IN NUMBER,
    P_INIT_MSG_LIST IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_COMMIT IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_VALIDATION_LEVEL IN NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
    X_RETURN_STATUS OUT VARCHAR2,
    X_MSG_COUNT OUT NUMBER,
    X_MSG_DATA OUT VARCHAR2,
    PIO_CDRBASEOBJCOLL IN OUT CDR_BASE_OBJ_COLL
);

**Parameters**  This API has standard parameters (see “Standard Parameters” on page 5) and the following parameter:

- **PIO_CDRBASEOBJCOLL** (Mandatory) This is a collection of CDR_BASE_OBJ_TYPES. For each Program that you want to remove, initialize a CDR_BASE_OBJ_TYPE and then extend the collection.
The following attributes are required: COMPANY_ID, OBJ_ID, OBJ_VER, OBJECT_ VERSION_NUMBER, NAMESPACE_OBJ_ID, NAMESPACE_OBJ_VER.

10.1.7 Create a Planned Output for a Log File

Use this API to create a log file Planned Output for a Program.

**Note:** You must define a log file Planned Output, which is part of the Program definition, through an instance of the Program.

**Name**  
CDR_PUB_DF_PROGRAM.CreateLogFilePlannedOutput

**Signature**

PROCEDURE CREATELOGFILEPLANNEDOUTPUT(
  P_API_VERSION IN NUMBER,
  P_INIT_MSG_LIST IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_COMMIT IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_VALIDATION_LEVEL IN NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
  X_RETURN_STATUS OUT VARCHAR2,
  X_MSG_COUNT OUT NUMBER,
  X_MSG_DATA OUT VARCHAR2,
  PIO_SOURCECODENAMING IN OUT CDR_NAMING_VERSION_OBJ_TYPE
);

**Parameters**

This API has standard parameters (see "Standard Parameters" on page 5) and the following parameter:

PIO_SOURCECODENAMING (Mandatory) This is a parameter of type cdr_naming_ version_obj_type that contains object attributes.

The required attributes are: COMPANY_ID, NAME, NAMESPACE_OBJ_ ID, NAMESPACE_OBJ_VER.

10.1.8 Assign a Planned Output

Use this API to assign a Planned Output to a Report Set Entry (RSE). If the Planned Output is already assigned to a different RSE, you must identify that RSE. This API then unassigns the Planned Output from the original RSE and reassigns it to the RSE you specify.

**Name**  
CDR_PUB_DF_PROGRAM.AssignPlandPOtoRSEntry

**Signature**

PROCEDURE ASSIGNPIANDPOTORSENTRY(
  P_API_VERSION IN NUMBER,
  P_INIT_MSG_LIST IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_COMMIT IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_VALIDATION_LEVEL IN NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
  X_RETURN_STATUS OUT VARCHAR2,
  X_MSG_COUNT OUT NUMBER,
  X_MSG_DATA OUT VARCHAR2,
  PI_RSECOMPANYID IN CDR_NAMINGS.COMPANY_ID%TYPE,
  PI_RSEOBJID IN CDR_NAMINGS.OBJ_ID%TYPE,
  PI_RSEOBJVER IN CDR_NAMING_VERSIONS.OBJ_VER%TYPE,
  PI_PIOBJID IN CDR_NAMINGS.OBJ_ID%TYPE,
  PI_POCOMPANYID IN CDR_NAMINGS.COMPANY_ID%TYPE,
PI_POOBJID IN CDR_NAMINGS.OBJ_ID%TYPE,
PI_ASGRSEOBJID IN CDR_NAMINGS.OBJ_ID%TYPE,
PI_ASGRSEOBJVER IN CDR_NAMING_VERSIONS.OBJ_VER%TYPE
);

Parameters  This API has standard parameters (see "Standard Parameters" on page 5) and the following parameters:

- **PI_RSECOMPANYID** (Mandatory) Enter the Company ID of the Report Set Entry to which you want to assign the Planned Output. To get your company ID, use CDR_DF_PUB_DEF_CONSTANTS.Current_Company_ID.
- **PI_RSEOBJID** (Mandatory) Enter the object ID of the Report Set Entry to which you want to assign the Planned Output.
- **PI_RSEOBJVER** (Mandatory) Enter the object version of the Report Set Entry to which you want to assign the Planned Output.
- **PIPIOBJID** (Mandatory) Enter the object ID of the Program instance to which the Planned Output belongs.
- **PI_POCOMPANYID** (Mandatory) Enter the Company ID of the Planned Output.
- **PI_POOBJID** (Mandatory) Enter the object ID of the Planned Output.
- **PI_ASGRSEOBJID** (Optional) If the Planned Output is already assigned to another RSE, enter the object ID of the other Report Set Entry.
- **PI_ASGRSEOBJVER** (Optional) If the Planned Output is already assigned to another RSE, enter the object version of the other Report Set Entry.

10.1.9 Modify a Manual Validation Flag Value

Use this API to change the value of the Program's manual validation (Force Validation Status to Development) flag.

Name  CDR_PUB_DF_PROGRAM. ModifyProgramDetails

Signature

PROCEDURE MODIFYPROGRAMDETAILS(
  P_API_VERSION IN NUMBER,
  P_INIT_MSG_LIST IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_COMMIT IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_VALIDATION_LEVEL IN NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
  X_RETURN_STATUS OUT VARCHAR2,
  X_MSG_COUNT OUT NUMBER,
  X_MSG_DATA OUT VARCHAR2,
  PIO_CDRNAMING IN OUT CDR_NAMING_VERSION_OBJ_TYPE,
  PI_PROGRAMOBJTYPE IN OUT CDR_PROGRAM_OBJ_TYPE
);

Parameters  This API has standard parameters (see "Standard Parameters" on page 5) and the following parameters:

- **PIO_CDRNAMING** (Mandatory) This is a parameter of table type CDR_NAMING_VERSION_OBJ_TYPE that contains object attributes. Enter values to identify the Program definition whose flag value you want to change.

  The following attributes are required: OBJ_ID, OBJ_VER, COMPANY_ID, NS_OBJ_ID, NS_OBJ_VER.
**PI_PROGRAMOBJTYPE** (Mandatory) This is a parameter of table type CDR PROGRAM_OBJ_TYPE that contains Program definition attributes.

The required attributes are OBJ_ID, OBJ_VER, COMPANY_ID. Enter values to identify the Program definition whose flag value you want to change and enter the new flag value in the attribute MANUAL_VALIDATION_FLAG_RC. Enter $yesno$yes for yes, and $yesno$no for no.
11 Report Sets

This section contains the following topics:

- Section 11.1, "Create and Modify Report Set Entries"
- Section 11.2, "Create and Modify Report Sets"
- Section 11.3, "Create and Modify Overlay Template Definitions"
- Section 11.4, "Report Set Overlay Template"

11.1 Create and Modify Report Set Entries

This is a public interface for Report Set Entry-related operations, including creating, modifying, and removing Report Set Entries, getting information about the Report Set and Report Set Entries, and changing the Report Set structure.

This section contains the following topics:

- Section 11.1.1, "Create a Report Set Entry"
- Section 11.1.2, "Add and Modify an Entry"
- Section 11.1.3, "Copy a Report Set Entry into Another"
- Section 11.1.4, "Modify a Report Set Entry"
- Section 11.1.5, "Move a Report Set Entry into Another"
- Section 11.1.6, "Reorder Report Set Entries in a Parent Report Set"
- Section 11.1.7, "Find if a Report Set is Checked Out"
- Section 11.1.8, "Check Unique and Strict Numbering in a Report Set"
- Section 11.1.9, "Identify if a Report Set Contains Child Entries"
- Section 11.1.10, "Find if a User has Modify Permission"
- Section 11.1.11, "Remove an Object from a Report Set Entry"
- Section 11.1.12, "Remove a Report Set Entry"
- Section 11.1.13, "Get a Report Set Name"
- Section 11.1.14, "Get a Title"
- Section 11.1.15, "Get a Chapter Number"
- Section 11.1.16, "Get a Parent Number"
- Section 11.1.17, "Get a List of Report Set Entry Titles"
- Section 11.1.18, "Get All RSE Titles in a Report Set"
11.1.1 Create a Report Set Entry

Use this API to create a Report Set Entry within a Report Set or another Report Set Entry.

**Name**  CDR_PUB_RS_REPORT_SET_ENTRY.CreateReportSetEntry

**Signature**

```sql
PROCEDURE CREATEREPORTSETENTRY(
    P_API_VERSION IN NUMBER,
    P_INIT_MSG_LIST IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_COMMIT IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_VALIDATION_LEVEL IN NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
    X_RETURN_STATUS OUT VARCHAR2,
    X_MSG_COUNT OUT NUMBER,
    X_MSG_DATA OUT VARCHAR2,
    PIO_SOURCECDRNAMING IN OUT CDR_NAMING_VERSION_OBJ_TYPE,
    PI_CDRRSENTRY IN OUT CDR_RS_ENTRY_OBJ_TYPE
);
```

**Parameters**  This API has standard parameters (see "Standard Parameters" on page 5) and the following parameters:

- **PIO_SOURCECDRNAMING** (Mandatory) This is a parameter of table type CDR_NAMING_VERSION_OBJ_TYPE that contains object attributes. Enter values for the new Report Set Entry. Use the NAMESPACE attributes to identify the Report Set or Report Set Entry in which you want to create the new Report Set Entry. For OBJECT_TYPE_RC enter $OBJTYPES$REPORTSETENTRY.

  The following attributes are required: COMPANY_ID,NAME,NAMESPACE_OBJ_ID,NAMESPACE_OBJ_VER.

- **PI_CDRRSENTRY** (Mandatory) This is a parameter of table type CDR_RS_ENTRY_OBJ_TYPE that contains attributes specific to Report Set Entries. Enter values for the new Report Set Entry.
The following attributes are required: COMPANY_ID, OBJ_ID, OBJ_VER, ENTRY_NUMBER, POSITION, UNIQUE_NUMBERING_FLAG_RC, STRICT_NUMBERING_FLAG_RC, PRE_NARRATIVE_ID, POST_NARRATIVE_ID, OMIT_FROM_INSTALL_FLAG_RC, TITLE, DELIMITER, PREFIX, POSTFIX, PLACEHOLDER_FLAG_RC, VOLUME_BREAK_FLAG_RC, PI_OBJ_ID, PO_COMPANY_ID, PO_OBJ_ID, ENTRY_NUMBER_FLAG_RC, PARENT_FLAG_RC, PREFIX_FLAG_RC, POSTFIX_FLAG_RC, DELIMITER_FLAG_RC.

11.1.2 Add and Modify an Entry

Use this API to add and/or modify one or more Report Set Entries within the same parent Report Set Entry or Report Set definition. For a new RSE to be added, the user can use any number for the OBJ_ID and OBJ_VER like 0 and a positive number.

**Name**  
CDR_PUB_RS_REPORT_SET_ENTRY.AddNewAndModifyEntries

**Signature**

```sql
PROCEDURE ADDNEWANDMODIFYENTRIES(
  P_API_VERSION IN NUMBER,
  P_INIT_MSG_LIST IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_COMMIT IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_VALIDATION_LEVEL IN NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
  X_RETURN_STATUS OUT VARCHAR2,
  X_MSG_COUNT OUT NUMBER,
  X_MSG_DATA OUT VARCHAR2,
  PI_CDRRSENAMEENTRYCOLL IN OUT CDR_RS_ENTRY_NAME_COLL,
  PIO_CDRRSECOLL IN OUT CDR_RS_ENTRY_OBJ_TYPE,
  COMPANYID IN CDR_NAMING_VERSIONS.COMPANY_ID%TYPE,
  NSOBJID IN CDR_NAMINGS.NAMESPACE_OBJ_ID%TYPE,
  NSOBJVER IN CDR_NAMING_VERSIONS.OBJ_VER%TYPE,
  OBJSUBTYPEID IN CDR_NAMINGS.OBJECT_SUBTYPE_ID%TYPE
);
```

**Parameters**  
This API has standard parameters (see “Standard Parameters” on page 5) and the following parameters:

- **PI_CDRRSENAMEENTRYCOLL** (Mandatory) This is a collection of CDR_RS_ENTRY_NAME_OBJs. For each Report Set Entry that you want to modify or add, initialize a CDR_RS_ENTRY_NAME_OBJ and then extend the collection.
  
  If you are modifying an RSE, enter its existing values.
  
  If you are creating a new RSE, enter 1 for OBJ_VER and enter a number for the OBJ_ID (it must be unique within the collection). The following attributes are required: NAME, ENTRY_NUMBER, OBJ_ID, OBJ_VER.

- **PIO_CDRRSECOLL** (Mandatory) This is a collection of CDR_RS_ENTRY_OBJ_TYPEs. For each Report Set Entry that you want to modify or add, initialize a CDR_RS_ENTRY_OBJ_TYPE (with new values for the attributes you want to change, if you are modifying existing Entries) and then extend the collection.
  
  The following attributes are required: COMPANY_ID, OBJ_ID, OBJ_VER, ENTRY_NUMBER, POSITION, UNIQUE_NUMBERING_FLAG_RC, STRICT_NUMBERING_FLAG_RC, PRE_NARRATIVE_ID, POST_NARRATIVE_ID, OMIT_FROM_INSTALL_FLAG_RC, TITLE, DELIMITER, PREFIX, POSTFIX, PLACEHOLDER_FLAG_RC, VOLUME_BREAK_FLAG_RC, PI_OBJ_ID, PO_COMPANY_ID, PO_OBJ_ID, ENTRY_NUMBER_FLAG_RC, PARENT_FLAG_RC, PREFIX_FLAG_RC, POSTFIX_FLAG_RC, DELIMITER_FLAG_RC.
COMPANYID (Mandatory) Enter your COMPANY_ID.

NSOBJID (Mandatory) Enter the OBJ_ID of the Report Set or Report Set Entry within which you want to add or modify Report Set Entries.

NSOBJVER (Mandatory) Enter the OBJ_VER of the Report Set or Report Set Entry within which you want to add or modify Report Set Entries.

OBJSUBTYPEID (Optional) If you are adding or modifying top-level Report Set Entries, enter the subtype ID of the Report Set.

If you are adding or modifying Report Set Entries contained in another Report Set Entry, leave this parameter null.

11.1.3 Copy a Report Set Entry into Another

Use this procedure to copy one or more Report Set Entries into another Report Set Entry. The target Report Set Entry may or may not belong to the same Report Set hierarchy.

Name    CDR_PUB_RS_REPORT_SET_ENTRY.CopyObjectsIntoReportSetEntry

Signature

PROCEDURE COPYOBJECTSINTOREPORTSETENTRY(
  P_API_VERSION IN    NUMBER,
  P_INIT_MSG_LIST IN    VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_COMMIT  IN    VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_VALIDATION_LEVEL  IN    NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
  X_RETURN_STATUS  OUT    VARCHAR2,
  X_MSG_COUNT  OUT    NUMBER,
  X_MSG_DATA  OUT    VARCHAR2,
  PIO_CDRBASEOBJCOLL  IN OUT    CDR_BASE_OBJ_COLL,
  PIO_TARGETBASEOBJ  IN OUT    CDR_BASE_OBJ_TYPE,
  RENUMBER  IN    VARCHAR2,
  PI_COPYPRGASSGNMT  IN    VARCHAR2
);

Parameters    This API has standard parameters (see "Standard Parameters" on page 5) and the following parameters:

- PIO_CDRBASEOBJCOLL (Mandatory) This is a collection of CDR_BASE_OBJ_TYPES.

  For each Report Set Entry that you want to copy into another Report Set Entry, initialize a CDR_BASE_OBJ_TYPE and then extend the collection.

  The following attributes are required: COMPANY_ID,OBJ_ID,OBJ_VER,OBJECT_VERSION_NUMBER,NAMESPACE_OBJ_ID,NAMESPACE_OBJ_VER.

- PIO_TARGETBASEOBJ (Mandatory) This is a parameter of table type CDR_BASE_OBJ_TYPE that contains object attributes. Enter values to identify the Report Set Entry into which you want to copy other Report Set Entries.
The following attributes are required: COMPANY_ID, OBJ_ID, OBJ_VER, OBJECT_VERSION_NUMBER, NAMESPACE_OBJ_ID, NAMESPACE_OBJ_VER

- **RENUMBER** (Mandatory) Enter 'Y' for copied Report Set Entries to appropriately numbered in their new location.
- **PI_COPYPRGASSGNMT** (Mandatory) When copying Report Set Entries, enter 'Y' to copy Program instances currently assigned to the Report Set Entries, their Planned Output assignments and mappings. Enter 'N' to avoid copying these Program instances in which case, all Planned Output assignments are lost.

### 11.1.4 Modify a Report Set Entry

Use this API to modify a Report Set Entry.

**Name** CDR_PUB_RS_REPORT_SET_ENTRY.ModifyReportSetEntry

**Signature**

```sql
PROCEDURE MODIFYREPORTSETENTRY(
    P_API_VERSION  IN    NUMBER,
    P_INIT_MSG_LIST  IN    VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_COMMIT  IN    VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_VALIDATION_LEVEL  IN    NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
    X_RETURN_STATUS  OUT    VARCHAR2,
    X_MSG_COUNT  OUT    NUMBER,
    X_MSG_DATA  OUT    VARCHAR2,
    PIO_SOURCECDRNAMING  IN OUT    CDR_NAMING_VERSION_OBJ_TYPE,
    PI_CDRRSENTRY  IN OUT    CDR_RS_ENTRY_OBJ_TYPE
);
```

**Parameters** This API has standard parameters (see "Standard Parameters" on page 5) and the following parameters:

- **PIO_SOURCECDRNAMING** (Mandatory) This is a parameter of table type CDR_NAMING_VERSIONS_OBJ_TYPE that contains CDR Naming Version attributes. Enter values to identify the Report Set Entry that you want modify and enter new values for the attributes you want to modify. You can change the name and description. All attributes are required.

- **PI_CDRRSENTRY** (Mandatory) This is a parameter of table type CDR_RS_ENTRY_OBJ_TYPE that contains object attributes. Enter values to identify the Report Set Entry that you want to modify and enter new values for the attributes you want to modify. You can change the Entry and Position numbers, the Strict and Unique Numbering flags, the Omit from Install flag, the Title, Delimiter, Prefix, Postfix, Placeholder flag and Volume Break flag values. All attributes are required.

### 11.1.5 Move a Report Set Entry into Another

Use this API to move one or more Report Set Entries into another Report Set Entry.

**Name** CDR_PUB_RS_REPORT_SET_ENTRY.MoveObjectsIntoReportSetEntry

**Signature**

```sql
PROCEDURE MOVEOBJECTSINTOREPORTSETENTRY{
    P_API_VERSION  IN    NUMBER,
    P_INIT_MSG_LIST  IN    VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_COMMIT  IN    VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
```

- **PIO_SOURCECDRNAMING**
Create and Modify Report Set Entries

P_VALIDATION_LEVEL IN NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
X_RETURN_STATUS OUT VARCHAR2,
X_MSG_COUNT OUT NUMBER,
X_MSG_DATA OUT VARCHAR2,
PIO_CDRBASEOBJCOLL IN OUT CDR_BASE_OBJ_COLL,
PIO_TARGETBASEOBJ IN OUT CDR_BASE_OBJ_TYPE,
RENUMBER IN VARCHAR2
);

Parameters
This API has standard parameters (see "Standard Parameters" on page 5) and the following parameters:

- **PIO_CDRBASEOBJCOLL** (Mandatory) This is a collection of CDR_BASE_OBJ_TYPES. For each Report Set Entry that you want to move, initialize a CDR_BASE_OBJ_TYPE and then extend the collection.

  The following attributes are required: COMPANY_ID,OBJ_ID,OBJ_VER,OBJECT_VERSION_NUMBER,NAMESPACE_OBJ_ID,NAMESPACE_OBJ_VER.

- **PIO_TARGETBASEOBJ** (Mandatory) This is a parameter of table type CDR_BASE_OBJ_TYPE that contains object attributes. Enter values to identify the Report Set or Report Set Entry into which you want to move Report Set Entries.

  The following attributes are required: COMPANY_ID,OBJ_ID,OBJ_VER,OBJECT_VERSION_NUMBER,NAMESPACE_OBJ_ID,NAMESPACE_OBJ_VER.

- **RENUMBER** Enter either TRUE or FALSE. When you move a Report Set Entry, the API determines whether or not the numbering of the target Report Set or Report Set Entry is still valid.

  If you enter TRUE, the API automatically renumbers the target to make its numbering valid.

  If you enter FALSE, the copy operation fails if it creates invalid numbering.

### 11.1.6 Reorder Report Set Entries in a Parent Report Set

Use this API to change the order of Report Set Entries within a parent Report Set or Report Set Entry.

**Name**  CDR_PUB_RS_REPORT_SET_ENTRY.ReorderReportSetEntry

**Signature**

PROCEDURE REORDERREPORTSETENTRY(
  P_API_VERSION IN NUMBER,
  P_INIT_MSG_LIST IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_COMMIT IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_VALIDATION_LEVEL IN NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
  X_RETURN_STATUS OUT VARCHAR2,
  X_MSG_COUNT OUT NUMBER,
  X_MSG_DATA OUT VARCHAR2,
  PI_REORDERCOLL IN OUT CDR_REORDER_OBJ_COLL
);

Parameters
This API has standard parameters (see "Standard Parameters" on page 5) and the following parameters:

- **PI_REORDERCOLL** (Mandatory) Initialize a CDR_REORDER_OBJ_TYPE for each child Report Set Entry in the same parent Report Set or Report Set Entry, including the
correct new Position and Entry Number, in the correct new order, and then extend the collection.

The following attributes are required: COMPANY_ID, OBJ_ID, OBJ_VER, OBJECT_VERSION_NUMBER, NAMESPACE_OBJ_ID, NAMESPACE_OBJ_VER, POSITION, ENTRY_NUMBER, OBJECT_VERSION_NUMBER.

### 11.1.7 Find if a Report Set is Checked Out

Use this API to determine whether or not the Report Set is currently checked out. The API returns `YESNO=YES` if it is checked out and `YESNO=NO` if it is not checked out.

**Name**  CDR_PUB_RS_REPORT_SET_ENTRY.GetRSCheckoutFlagRC

**Signature**

FUNCTION GETRSCHECKOUTFLAGRC(
    P_API_VERSION  IN    NUMBER,
    P_INIT_MSG_LIST  IN    VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_COMMIT  IN    VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_VALIDATION_LEVEL  IN    NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
    PI_COMPANYID  IN    CDR_NAMINGS.COMPANY_ID%TYPE,
    PI_OBJID  IN    CDR_NAMINGS.OBJ_ID%TYPE
) RETURN CDR_NAMINGS.CHECKED_OUT_FLAG_RC%TYPE;

**Return**  CDR_NAMINGS.CHECKED_OUT_FLAG_RC%TYPE

**Description** varchar, returns 'YESNO=YES' or 'YESNO=NO'

**Parameters**  This API has standard parameters (see "Standard Parameters" on page 5) and the following parameters:

- **PI_COMPANYID** (Mandatory) Enter the COMPANY_ID of the Report Set Entry.
- **PI_OBJID** (Mandatory) Enter the OBJ_ID of the Report Set Entry.

### 11.1.8 Check Unique and Strict Numbering in a Report Set

Use this API to check whether a Report Set’s Entries conform to the rules you specify for unique and strict numbering.

**Name**  CDR_PUB_RS_REPORT_SET_ENTRY.IsValidChildrenRSEntries

**Signature**

FUNCTION ISVALIDCHILDRENRSENTRIES(
    P_API_VERSION IN    NUMBER,
    P_INIT_MSG_LIST IN    VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_COMMIT IN    VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_VALIDATION_LEVEL IN    NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
    COMPANYID IN CDR_NAMINGS.COMPANY_ID%TYPE,
    OBJID IN CDR_NAMINGS.OBJ_ID%TYPE,
    OBJVER IN CDR_NAMING_VERSIONS.OBJ_VER%TYPE,
    UNIQUEFLAG IN CDR_RS_ENTRIES.UNIQUE_NUMBERING_FLAG_RC%TYPE,
    STRICTFLAG IN CDR_RS_ENTRIES.STRICT_NUMBERING_FLAG_RC%TYPE
) RETURN VARCHAR2;

**Return**  Type VARCHAR2

**Description** varchar true or false
**Parameters**  This API has standard parameters (see "Standard Parameters" on page 5) and the following parameters:

- **COMPANYID** Enter the Company ID of the Report Set.
- **OBJID** Enter the object ID of the Report Set.
- **OBJVER** Enter the object version of the Report Set.
- **UNIQUEFLAG** Enter Y to validate the Report Set’s numbering for uniqueness. Enter N to allow non-unique Report Set Entry numbers within the Report Set.
- **STRICTFLAG** Enter Y to validate the Report Set for strictly sequential numbering, with no gaps allowed. Enter N to allow gaps in numbering between sibling Report Set Entries within the Report Set.

### 11.1.9 Identify if a Report Set Contains Child Entries

Use this API to determine whether or not a Report Set or Report Set Entry contains child Report Set Entries.

**Name**  CDR_PUB_RS_REPORT_SET_ENTRY.HasChildren

**Signature**

FUNCTION HASCHILDREN(
  P_API_VERSION IN NUMBER,
  P_INIT_MSG_LIST IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_COMMIT IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_VALIDATION_LEVEL IN NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
  PI_COMPANYID IN CDR_NAMINGS.COMPANY_ID%TYPE,
  PI_NS_OBJID IN CDR_NAMINGS.OBJ_ID%TYPE,
  PI_NS_OBJVER IN CDR_NAMING_VERSIONS.OBJ_VER%TYPE
) RETURN VARCHAR2;

**Return**  Type VARCHAR2

Description varchar Y or N depending on if the RS has children or not.

**Parameters**  This API has standard parameters (see "Standard Parameters" on page 5) and the following parameters:

- **PI_COMPANYID** (Mandatory) Enter the COMPANY_ID of the Report Set or Report Set Entry.
- **PI_NS_OBJID** (Mandatory) Enter the OBJ_ID of the Report Set or Report Set Entry.
- **PI_NS_OBJVER** (Mandatory) Enter the OBJ_VER of the Report Set or Report Set Entry.

### 11.1.10 Find if a User has Modify Permission

Use this API to determine whether or not you (the current user) have permission to modify a particular Report Set Entry. This API takes into consideration the validation status of the Report Set or RSE and whether you have the RS Modify privilege, the RS Modify QC privilege, the RS Modify Production privilege, or none of these privileges.

**Name**  CDR_PUB_RS_REPORT_SET_ENTRY.HasReportModPermission

**Signature**
FUNCTION HASREPORTMODPERMISSION(
    P_API_VERSION  IN    NUMBER,
    P_INIT_MSG_LIST IN    VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_COMMIT IN    VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_VALIDATION_LEVEL IN    NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
    PI_COMPANYID IN    CDR_NAMINGS.COMPANY_ID%TYPE,
    PI_OBJID IN    CDR_NAMINGS.OBJ_ID%TYPE,
    PI_OBJVER IN    CDR_NAMING_VERSIONS.OBJ_VER%TYPE,
    RAISEEXECPTION IN    VARCHAR2 := 'N'
) RETURN VARCHAR2;

Return  Type VARCHAR2
Description varchar2

Parameters  This API has standard parameters (see “Standard Parameters” on page 5) and the following parameters:

- **PI_COMPANYID** (Mandatory) Enter the COMPANY_ID of the Report Set Entry.
- **PI_OBJID** (Mandatory) Enter the OBJ_ID of the Report Set Entry.
- **PI_OBJVER** (Mandatory) Enter the OBJ_VER of the Report Set Entry.
- **RAISEEXECPTION** If you enter Y here and you do not have the required privileges to modify the Report Set Entry, the API raises an exception.

If you enter N, the API tests for different conditions.

If you do not have the required privileges to modify the RSE, the API returns N irrespective of the RSE’s validation status.

If the RSE’s validation status is Development, the API returns Y if you have modify privileges on the RSE and N if you do not.

If the RSE’s validation status is QC or Production, the API checks if you have Modify QC or Modify Production, respectively.

If you do not have the privilege required to modify an RSE of the current validation status, the API returns VIRTUAL_LOCK.

However, you may be able to modify other RSEs in the same Report Set.

### 11.1.11 Remove an Object from a Report Set Entry

Use this API to remove one or more Report Set Entries from a Report Set or parent Report Set Entry.

Name  CDR_PUB_RS_REPORT_SET_ENTRY.RemoveObjectsFromRSEntries

Signature

PROCEDURE REMOVEOBJECTSFROMRSENTRIES(
    P_API_VERSION  IN    NUMBER,
    P_INIT_MSG_LIST IN    VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_COMMIT IN    VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_VALIDATION_LEVEL IN    NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
    X_RETURN_STATUS  OUT    VARCHAR2,
    X_MSG_COUNT  OUT    NUMBER,
    X_MSG_DATA  OUT    VARCHAR2,
    PIO_CDRBASEOBJCOLL IN OUT    CDR_BASE_OBJ_COLL,
    RENUMBER IN    VARCHAR2
);
Parameters  This API has standard parameters (see "Standard Parameters" on page 5) and the following parameters:

- **PIO_CDRBASEOBJCOLL** (Mandatory) This is a collection of CDR_BASE_OBJ_TYPEs. For each Report Set Entry that you want to copy, initialize a CDR_BASE_OBJ_TYPE and then extend the collection.
  The following attributes are required: COMPANY_ID, OBJ_ID, OBJ_VER, OBJECT_VERSION_NUMBER, NAMESPACE_OBJ_ID, NAMESPACE_OBJ_VER.

- **RENUMBER** Enter either TRUE or FALSE. When you remove a Report Set Entry, the API determines whether or not the numbering of the parent Report Set or Report Set Entry is still valid. If you enter TRUE, the API automatically rennumbers the target to make its numbering valid. If you enter FALSE, the Remove operation fails if it creates invalid numbering.

### 11.1.12 Remove a Report Set Entry

Use this API to remove a Report Set Entry from a Report Set. If the removal of the RSE causes invalid numbering in the remaining RSEs, the API generates an error.

**Name**  CDR_PUB_RS_REPORT_SET_ENTRY.RemoveReportSetEntry

**Signature**

```sql
PROCEDURE REMOVERTSETENTRY(
    P_API_VERSION  IN NUMBER,
    P_INIT_MSG_LIST  IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_COMMIT  IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_VALIDATION_LEVEL  IN NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
    X_RETURN_STATUS  OUT VARCHAR2,
    X_MSG_COUNT  OUT NUMBER,
    X_MSG_DATA  OUT VARCHAR2,
    PIO_CDRBASEOBJ  IN OUT CDR_BASE_OBJ_TYPE
);
```

Parameters  This API has standard parameters (see "Standard Parameters" on page 5) and the following parameter:

- **PIO_CDRBASEOBJ** (Mandatory) This is a parameter of table type CDR_BASE_OBJ_TYPE that contains object attributes. Enter values to identify the Report Set Entry from which you want to unassign the Planned Output.

  The following attributes are required: COMPANY_ID, OBJ_ID AND OBJ_VER, NAMESPACE_OBJ_ID, NAMESPACE_OBJ_VER, OBJECT_VERSION_NUMBER.

### 11.1.13 Get a Report Set Name

Use this API to get the name of the Report Set you specify.

**Name**  CDR_PUB_RS_REPORT_SET_ENTRY.GetReportSetName

**Signature**

```sql
FUNCTION GETREPORTSETNAME(
    P_API_VERSION  IN NUMBER,
    P_INIT_MSG_LIST  IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_COMMIT  IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_VALIDATION_LEVEL  IN NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
    PI_COMPANYID  IN CDR_NAMINGS.COMPANY_ID%TYPE,
    X_RETURN_STATUS  OUT VARCHAR2,
    X_MSG_COUNT  OUT NUMBER,
    X_MSG_DATA  OUT VARCHAR2,
    PI_CDRBASEOBJ  IN OUT CDR_BASE_OBJ_TYPE
)
```
PI_OBJID IN CDR_NAMINGS.OBJ_ID%TYPE
) RETURN CDR_NAMINGS.NAME%TYPE;

Return Type CDR_NAMINGS.NAME%TYPE
Description varchar Name of the RS

Parameters This API has standard parameters (see "Standard Parameters" on page 5) and the following parameters:

- **PI_COMPANYID** (Mandatory) Enter the Company ID of the Report Set.
- **PI_OBJID** (Mandatory) Enter the Object ID of the Report Set.

### 11.1.14 Get a Title

Use this API to get the full title of a Report Set or Report Set Entry. If you do not specify a version, the API returns the value for the most recent version.

**Name** CDR_PUB_RS_REPORT_SET_ENTRY.GetTitle

**Signature**

FUNCTION GETTITLE(
    P_API_VERSION  IN    NUMBER,
    P_INIT_MSG_LIST  IN    VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_COMMIT  IN    VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_VALIDATION_LEVEL  IN    NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
    PI_COMPANYID  IN    CDR_NAMINGS.COMPANY_ID%TYPE,
    PI_OBJID  IN    CDR_NAMINGS.OBJ_ID%TYPE,
    PI_OBJVER  IN    CDR_NAMING_VERSIONS.OBJ_VER%TYPE
) RETURN VARCHAR2;

Return Type VARCHAR2
Description overcharge returns the title of the RS or RSE

Parameters This API has standard parameters (see "Standard Parameters" on page 5) and the following parameters:

- **PI_COMPANYID** (Mandatory) Enter the COMPANY_ID of the Report Set Entry.
- **PI_OBJID** (Mandatory) Enter the OBJ_ID of the Report Set Entry.
- **PI_OBJVER** (Mandatory) Enter the OBJ_VER of the Report Set Entry.

### 11.1.15 Get a Chapter Number

Use this API to get the chapter number of a Report Set Entry.

**Name** CDR_PUB_RS_REPORT_SET_ENTRY.GetChapterNumber

**Signature**

FUNCTION GETCHAPTERNUMBER(
    P_API_VERSION  IN    NUMBER,
    P_INIT_MSG_LIST  IN    VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_COMMIT  IN    VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_VALIDATION_LEVEL  IN    NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
    PI_COMPANYID  IN    CDR_NAMINGS.COMPANY_ID%TYPE,
    PI_OBJID  IN    CDR_NAMINGS.OBJ_ID%TYPE,
    PI_OBJVER  IN    CDR_NAMING_VERSIONS.OBJ_VER%TYPE
) RETURN VARCHAR2;
### 11.1.16 Get a Parent Number

Use this API to get the chapter number of the parent Report Set Entry of the RSE you specify. If the parent is the Report Set itself, the API returns Null.

**Name**  
CDR_PUB_RS_REPORT_SET_ENTRY.GetParentNumber

**Signature**

FUNCTION GETPARENTNUMBER(
    P_API_VERSION IN NUMBER,
    P_INIT_MSG_LIST IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_COMMIT IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_VALIDATION_LEVEL IN NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
    PI_COMPANYID IN CDR_NAMINGS.COMPANY_ID%TYPE,
    PI_OBJID IN CDR_NAMINGS.OBJ_ID%TYPE,
    PI_OBJVER IN CDR_NAMING_VERSIONS.OBJ_VER%TYPE
) RETURN VARCHAR2;

**Return**  
Type VARCHAR2

Description: varchar returns the parent number of the object

**Parameters**  
This API has standard parameters (see "Standard Parameters" on page 5) and the following parameters:

- **PI_COMPANYID** (Mandatory) Enter the COMPANY_ID of the Report Set Entry.
- **PI_OBJID** (Mandatory) Enter the OBJ_ID of the Report Set Entry.
- **PI_OBJVER** (Mandatory) Enter the OBJ_VER of the Report Set Entry.

### 11.1.17 Get a List of Report Set Entry Titles

Use this API to get a list of the full titles and version numbers of all the RSEs in a direct path to a particular Report Set Entry, from the top of the Report Set down to the Report Set Entry you specify.

**Name**  
CDR_PUB_RS_REPORT_SET_ENTRY.TopDownListOfRSetitles

**Signature**

FUNCTION TOPDOWNLISTOFRSETITLES(
    P_API_VERSION IN NUMBER,
    P_INIT_MSG_LIST IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_COMMIT IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_VALIDATION_LEVEL IN NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
PI_NCOMPANYID IN CDR_NAMINGS.COMPANY_ID%TYPE,
PI_NRSEOBJID IN CDR_NAMINGS.OBJ_ID%TYPE,
PI_NRSEOBJVER IN CDR_NAMING_VERSIONS.OBJ_VER%TYPE,
PI_NRSOBJID IN CDR_NAMINGS.OBJ_ID%TYPE,
PI_NRSOBJVER IN CDR_NAMING_VERSIONS.OBJ_VER%TYPE)
) RETURN CDR_VALS_COLL;

Return Type CDR_VALS_COLL
Description CDR_VALS_COLL returns the list of RSE titles top down

Parameters This API has standard parameters (see "Standard Parameters" on page 5) and the following parameters:
- **PI_NCOMPANYID** (Mandatory) Enter the COMPANY_ID of the Report Set Entry.
- **PI_NCOMPANYID** (Mandatory) Enter the OBJ_ID of the Report Set Entry.
- **PI_NRSEOBJID** (Mandatory) Enter the OBJ_ID of the Report Set Entry.
- **PI_NRSOBJID** (Mandatory) Enter the OBJ_ID of the parent Report Set.
- **PI_NRSOBJVER** (Mandatory) Enter the OBJ_VER of the Report Set Entry.

11.1.18 Get All RSE Titles in a Report Set
Use this API to get a list of the full titles of all the RSEs in a Report Set.

Name CDR_PUB_RS_REPORT_SET_ENTRY.TopDownListOfRSeTitles

Signature

FUNCTION TOPDOWNLISTOFRSETITLES(
  P_API_VERSION IN NUMBER,
  P_INIT_MSG_LIST IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_COMMIT IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_VALIDATION_LEVEL IN NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
  PI_COMPANYID IN CDR_NAMINGS.COMPANY_ID%TYPE,
  PI_OBJID IN CDR_NAMINGS.OBJ_ID%TYPE,
  PI_OBJVER IN CDR_NAMING_VERSIONS.OBJ_VER%TYPE
) RETURN CDR_VALS_COLL;

Return Type CDR_VALS_COLL
Description CDR_VALS_COLL returns the list of RSE titles top down

Parameters This API has standard parameters (see "Standard Parameters" on page 5) and the following parameters:
- **PI_COMPANYID** (Mandatory) Enter the COMPANY_ID of the Report Set Entry.
- **PI_OBJID** (Mandatory) Enter the OBJ_ID of the Report Set Entry.
- **PI_OBJVER** (Mandatory) Enter the OBJ_VER of the Report Set Entry.

11.1.19 Get Attribute Values Derived from a Parent
Use this API to retrieve attribute values for a Report Set Entry that are derived from its parent Report Set or Report Set Entry, including those that are dynamically derived and those that are derived only when the child Report Set Entry is initially created.
Create and Modify Report Set Entries

Name  CDR_PUB_RS_REPORT_SET_ENTRY.GetDerivedAttrValuesFromParent

Signature

PROCEDURE GETDERIVEDATTRVALUESFROMPARENT(
    P_API_VERSION IN NUMBER,
    P_INIT_MSG_LIST IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_COMMIT IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_VALIDATION_LEVEL IN NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
    X_RETURN_STATUS OUT VARCHAR2,
    X_MSG_COUNT OUT NUMBER,
    X_MSG_DATA OUT VARCHAR2,
    PI_COMPANY_ID IN CDR_NAMINGS.COMPANY_ID%TYPE,
    PI_NAMESPACE_OBJ_ID IN CDR_NAMINGS.NAMESPACE_OBJ_ID%TYPE,
    PI_NAMESPACE_OBJ_VER IN CDR_NAMING_VERSIONS.NAMESPACE_START_OBJ_VER%TYPE,
    PI_CDRRSENTRY IN OUT CDR_RS_ENTRY_OBJ_TYPE,
    PI_PARENT_NUMBER OUT VARCHAR2);

Parameters  This API has standard parameters (see "Standard Parameters" on page 5) and the following parameters:

- **PI_COMPANY_ID** (Mandatory) Enter the COMPANY_ID of the parent Report Set or Report Set Entry.
- **PI_NAMESPACE_OBJ_ID** (Mandatory) Enter the OBJ_ID of the Report Set or Report Set Entry.
- **PI_NAMESPACE_OBJ_VER** (Mandatory) Enter the OBJ_VER of the parent Report Set or Report Set Entry.
- **PI_CDRRSENTRY** (Mandatory) This is a parameter of table type CDR_RS_ENTRY_OBJ_TYPE that contains attributes specific to Report Set Entries. Enter values to identify the Report Set Entry for which you want to retrieve inherited attributes.
- **PI_PARENT_NUMBER** This output parameter returns the chapter number of the parent Report Set Entry.
  
  If the parent is the Report Set itself, the parameter returns NULL.

11.1.20  Get the Lowest Entry Number

Use this API to get the number of the lowest numbered child Report Set Entry in the Report Set or RSE you specify.

Name  CDR_PUB_RS_REPORT_SET_ENTRY.GetMinEntryNumber

Signature

FUNCTION GETMINENTRYNUMBER(
    P_API_VERSION IN NUMBER,
    P_INIT_MSG_LIST IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_COMMIT IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_VALIDATION_LEVEL IN NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
    PI_COMPANY_ID IN CDR_NAMINGS.COMPANY_ID%TYPE,
    PI_NAMESPACE_OBJ_ID IN CDR_NAMINGS.NAMESPACE_OBJ_ID%TYPE,
    PI_NAMESPACE_OBJ_VER IN CDR_NAMING_VERSIONS.NAMESPACE_START_OBJ_VER%TYPE
) RETURN CDR_RS_ENTRIES.ENTRY_NUMBER%TYPE;

Return  Type CDR_RS_ENTRIES.ENTRY_NUMBER%TYPE
Description Minimum of the number of the RSEs in the report set

**Parameters** This API has standard parameters (see "Standard Parameters" on page 5) and the following parameters:

- **PI_COMPANY_ID** (Mandatory) Enter the COMPANY_ID of the Report Set or Report Set Entry.
- **PI_NAMESPACE_OBJ_ID** (Mandatory) Enter the OBJ_ID of the Report Set.
- **PI_NAMESPACE_OBJ_VER** (Mandatory) Enter the OBJ_VER of the Report Set.

11.1.21 Get the Total Number of Report Set Entries

Use this API to get the number of the highest numbered child Report Set Entry in the Report Set or RSE you specify.

**Name** CDR_PUB_RS_REPORT_SET_ENTRY.GetMaxEntryNumber

**Signature**

FUNCTION GETMAXENTRYNUMBER(  
P_API_VERSION IN NUMBER,  
P_INIT_MSG_LIST IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,  
P_COMMIT IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,  
P_VALIDATION_LEVEL IN NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,  
PI_COMPANY_ID IN CDR_NAMINGS.COMPANY_ID%TYPE,  
PI_NAMESPACE_OBJ_ID IN CDR_NAMINGS.NAMESPACE_OBJ_ID%TYPE,  
PI_NAMESPACE_OBJ_VER IN CDR_NAMING_VERSIONS.NAMESPACE_START_OBJ_VER%TYPE  ) RETURN CDR_RS_ENTRIES.ENTRY_NUMBER%TYPE;

**Return** Type CDR_RS_ENTRIES.ENTRY_NUMBER%TYPE

Description Maximum of the number of the RSEs in the report set

**Parameters** This API has standard parameters (see "Standard Parameters" on page 5) and the following parameters:

- **PI_COMPANY_ID** (Mandatory) Enter the COMPANY_ID of the Report Set.
- **PI_NAMESPACE_OBJ_ID** (Mandatory) Enter the OBJ_ID of the Report Set.
- **PI_NAMESPACE_OBJ_VER** (Mandatory) Enter the OBJ_VER of the Report Set.

11.1.22 Create a Narrative

Use this API to create a narrative for an existing Report Set Entry.

**Name** CDR_PUB_RS_REPORT_SET_ENTRY.CreateNarrative

**Signature**

PROCEDURE CREATENARRATIVE(  
P_API_VERSION IN NUMBER,  
P_INIT_MSG_LIST IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,  
P_COMMIT IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,  
P_VALIDATION_LEVEL IN NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,  
X_RETURN_STATUS OUT VARCHAR2,  
X_MSG_COUNT OUT NUMBER,  
X_MSG_DATA OUT VARCHAR2,  
PI_CDRRSENARRATIVE IN OUT CDR_RSE_NARRATIVE_OBJ_TYPE,  
PIO_BASEOBJECT IN OUT CDR_BASE_OBJ_TYPE
Parameters  This API has standard parameters (see "Standard Parameters" on page 5) and the following parameters:

- **PI_CDRRSENARRATIVE** (Mandatory) This is a parameter of type CDR_RSE_NARRATIVE_OBJ_TYPE that contains details of the narrative to be added.

  The following attributes are required: COMPANY_ID, NARRATIVE_MODE(PRE/POST), NARRATIVE_TEXT, NARRATIVE_TYPE(TEXT or TEXT/PLAIN), FILE_NAME.

- **PIO_BASEOBJECT** (Mandatory) This is a parameter of table type CDR_BASE_OBJ_TYPE that contains object attributes. Enter values to identify the Report Set Entry to which you want to create a Narrative.

  The following attributes are required: COMPANY_ID, OBJ_ID AND OBJ_VER, NAMESPACE_OBJ_ID, NAMESPACE_OBJ_VER, OBJECT_VERSION_NUMBER.

### 11.1.23 Update a Narrative

Use this API to update a narrative for an existing Report Set Entry.

**Name**  CDR_PUB_RS_REPORT_SET_ENTRY.UpdateNarrative

**Signature**

PROCEDURE UPDATENARRATIVE(  
P_API_VERSION IN NUMBER,  
P_INIT_MSG_LIST IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,  
P_COMMIT IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,  
P_VALIDATION_LEVEL IN NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,  
X_RETURN_STATUS OUT VARCHAR2,  
X_MSG_COUNT OUT NUMBER,  
X_MSG_DATA OUT VARCHAR2,  
PI_CDRRSENARRATIVE IN OUT CDR_RSE_NARRATIVE_OBJ_TYPE,  
PIO_BASEOBJECT IN OUT CDR_BASE_OBJ_TYPE  
);  

Parameters  This API has standard parameters (see "Standard Parameters" on page 5) and the following parameters:

- **PI_CDRRSENARRATIVE** (Mandatory) This is a parameter of type CDR_RSE_NARRATIVE_OBJ_TYPE that contains details of the narrative to be updated.

  The following attributes are required: COMPANY_ID, NARRATIVE_MODE(PRE/POST), NARRATIVE_TEXT, NARRATIVE_TYPE(TEXT or TEXT/PLAIN), FILE_NAME.

- **PIO_BASEOBJECT** (Mandatory) This is a parameter of table type CDR_BASE_OBJ_TYPE that contains object attributes. Enter values to identify the Report Set Entry from which you want to update a narrative.

  The following attributes are required: COMPANY_ID, OBJ_ID AND OBJ_VER, NAMESPACE_OBJ_ID, NAMESPACE_OBJ_VER, OBJECT_VERSION_NUMBER.

### 11.1.24 Delete a Narrative

Use this API to delete a narrative for an existing Report Set Entry.
Name  CDR_PUB_RS_REPORT_SET_ENTRY.DeleteNarrative

Signature

PROCEDURE DELETENARRATIVE(
    P_API_VERSION  IN    NUMBER,
    P_INIT_MSG_LIST  IN    VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_COMMIT  IN    VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_VALIDATION_LEVEL  IN    NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
    X_RETURN_STATUS  OUT    VARCHAR2,
    X_MSG_COUNT  OUT    NUMBER,
    X_MSG_DATA  OUT    VARCHAR2,
    PI_CDRRSENARRATIVE  IN OUT    CDR_RSE_NARRATIVE_OBJ_TYPE,
    PIO_BASEOBJECT  IN OUT    CDR_BASE_OBJ_TYPE
);

Parameters  This API has standard parameters (see "Standard Parameters" on page 5) and the following parameters:

- **PI_CDRRSENARRATIVE** (Mandatory) This is a parameter of type CDR_RSE_NARRATIVE_OBJ_TYPE that contains details of the narrative to be deleted.
  
  The following attributes are required: COMPANY_ID, NARRATIVE_MODE(PRE/POST).

- **PIO_BASEOBJECT** (Mandatory) This is a parameter of table type CDR_BASE_OBJ_TYPE that contains object attributes. Enter values to identify the Report Set Entry from which you want to delete a narrative.
  
  The following attributes are required: COMPANY_ID, OBJ_ID AND OBJ_VER,NAMESPACE_OBJ_ID,NAMESPACE_OBJ_VER, OBJECT_VERSION_NUMBER.

11.1.25  Check if Copying Retains Valid Numbering in a Target Report Set

Use this API to determine whether copying a given set of Report Set Entries into another Report Set or Report Set Entry would result in invalid numbering in the target RS or RSE.

Name  CDR_PUB_RS_REPORT_SET_ENTRY.ValidateCopyRSEList

Signature

FUNCTION VALIDATECOPYRSELIST(
    P_API_VERSION  IN    NUMBER,
    P_INIT_MSG_LIST  IN    VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_COMMIT  IN    VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_VALIDATION_LEVEL  IN    NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
    PIO_CDRBASEOBJCOLL  IN    CDR_BASE_OBJ_COLL,
    PIO_TARGETBASEOBJ  IN    CDR_BASE_OBJ_TYPE
) RETURN VARCHAR2;

Return  Type VARCHAR2

Description varchar2 success or failure

Parameters  This API has standard parameters (see "Standard Parameters" on page 5) and the following parameters:
11.1.26 Check if a Move Retains Valid Numbering in a Target Report Set

Use this API to determine whether moving a given set of Report Set Entries into another Report Set or Report Set Entry would result in invalid numbering in the target RS or RSE. The API returns TRUE if the numbering will remain valid or FALSE if the numbering will become invalid.

Name: CDR_PUB_RS_REPORT_SET_ENTRY.ValidateMoveIntoRSEList

Signature:
FUNCTION VALIDATEMOVEINTORSELIST(
  P_API_VERSION IN NUMBER,
  P_INIT_MSG_LIST IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_COMMIT IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_VALIDATION_LEVEL IN NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
  PIO_CDRBASEOBJCOLL IN CDR_BASE_OBJ_COLL,
  PIO_TARGETBASEOBJ IN CDR_BASE_OBJ_TYPE
) RETURN VARCHAR2;

Return: Type VARCHAR2
Description: varchar2 success or failure
Parameters: This API has standard parameters (see "Standard Parameters" on page 5) and the following parameters:

- PIO_CDRBASEOBJCOLL (Mandatory) This is a collection of CDR_BASE_OBJ_TYPES. For each Report Set Entry that you want to copy, initialize a CDR_BASE_OBJ_TYPE and then extend the collection.

  The following attributes are required: COMPANY_ID,OBJ_ID,OBJ_VER,OBJECT_VERSION_NUMBER,NAMESPACE_OBJ_ID,NAMESPACE_OBJ_VER.

- PIO_TARGETBASEOBJ (Mandatory) This is a parameter of table type CDR_BASE_OBJ_TYPE that contains object attributes. Enter values to identify the Report Set or Report Set Entry into which you want to copy Report Set Entries.

  The following attributes are required: COMPANY_ID,OBJ_ID,OBJ_VER,OBJECT_VERSION_NUMBER,NAMESPACE_OBJ_ID,NAMESPACE_OBJ_VER.

11.1.27 Check if a Move Retains Valid Numbering in the Parent Report Set

Use this API to determine whether moving a given set of Report Set Entries out of its parent Report Set or RSE would result in invalid numbering in the parent RS or RSE.

Name: CDR_PUB_RS_REPORT_SET_ENTRY.ValidateMoveFromRSEList
Create and Modify Report Set Entries

Signature
FUNCTION VALIDATEMOVEFROMRSELIST(
  P_API_VERSION  IN    NUMBER,
  P_INIT_MSG_LIST  IN    VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_COMMIT  IN    VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_VALIDATION_LEVEL  IN    NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
  PIO_CDRBASEOBJCOLL  IN    CDR_BASE_OBJ_COLL
) RETURN VARCHAR2;

Return  Type VARCHAR2
Description varchar2 success or failure

Parameters  This API has standard parameters (see "Standard Parameters" on page 5) and the following parameter:

PIO_CDRBASEOBJCOLL (Mandatory) This is a collection of CDR_BASE_OBJ_TYPES. For each Report Set Entry that you want to copy, initialize a CDR_BASE_OBJ_TYPE and then extend the collection.

The following attributes are required: COMPANY_ID,OBJ_ID,OBJ_VER,OBJECT_VERSION_NUMBER,NAMESPACE_OBJ_ID,NAMESPACE_OBJ_VER.

11.1.28 Check if Removal Retains Valid Numbering in a Parent Report Set
Use this API to determine whether removing a given set of Report Set Entries would result in invalid numbering in the parent RS or RSE.

Name  CDR_PUB_RS_REPORT_SET_ENTRYCDR_PUB_RS_REPORT_SETENTRY.ValidateRemoveRSEList

Signature
FUNCTION VALIDATEREMOVERSELIST(
  P_API_VERSION  IN    NUMBER,
  P_INIT_MSG_LIST  IN    VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_COMMIT  IN    VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_VALIDATION_LEVEL  IN    NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
  PIO_CDRBASEOBJCOLL  IN    CDR_BASE_OBJ_COLL
) RETURN VARCHAR2;

Return  Type VARCHAR2
Description varchar2 success or failure

Parameters  This API has standard parameters (see "Standard Parameters" on page 5) and the following parameter:

PIO_CDRBASEOBJCOLL (Mandatory) This is a collection of CDR_BASE_OBJ_TYPES. For each Report Set Entry that you want to copy, initialize a CDR_BASE_OBJ_TYPE and then extend the collection.

The following attributes are required: COMPANY_ID,OBJ_ID,OBJ_VER,OBJECT_VERSION_NUMBER,NAMESPACE_OBJ_ID,NAMESPACE_OBJ_VER.

11.1.29 Check if Reordering Retains Valid Numbering in a Parent Report Set
Use this API to determine whether reordering a collection of Report Set Entries would result in invalid numbering in the parent RS or RSE. To actually reorder the RSEs use ReorderReportSetEntry.
Create and Modify Report Sets

Name   CDR_PUB_RS_REPORT_SET_ENTRY.ValidateReorderList

Signature

FUNCTION VALIDATEREORDERLIST(
    P_API_VERSION IN    NUMBER,
    P_INIT_MSG_LIST IN    VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_COMMIT IN    VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_VALIDATION_LEVEL IN    NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
    REORDERCOLL IN    CDR_REORDER_OBJ_COLL
) RETURN VARCHAR2

Return   Type VARCHAR2
Description varchar2 success or failure

Parameters   This API has standard parameters (see "Standard Parameters" on page 5) and the following parameter:

REORDERCOLL Initialize a CDR_REORDER_OBJ_TYPE for each child Report Set Entry in the same parent Report Set or Report Set Entry, including the correct new position and Entry Number, in the correct new order, and then extend the collection. All the attributes are mandatory.

11.1.30 Unassign a Planned Output

Use this API to unassign a Planned Output from a Report Set Entry.

Name   CDR_PUB_RS_REPORT_SET_ENTRY.RemovePOAssignment

Signature

PROCEDURE REMOVEPOASSIGNMENT(
    P_API_VERSION IN    NUMBER,
    P_INIT_MSG_LIST IN    VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_COMMIT IN    VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_VALIDATION_LEVEL IN    NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
    X_RETURN_STATUS  OUT    VARCHAR2,
    X_MSG_COUNT  OUT    NUMBER,
    X_MSG_DATA  OUT    VARCHAR2,
    PI_CDRBASEOBJ  IN    CDR_BASE_OBJ_TYPE
);

Parameters   This API has standard parameters (see "Standard Parameters" on page 5) and the following parameter:

PI_CDRBASEOBJ (Mandatory) This is a parameter of table type CDR_BASE_OBJ_TYPE that contains object attributes. Enter values to identify the Report Set Entry from which you want to unassign the Planned Output.

The following attributes are required: COMPANY_ID, OBJ_ID AND OBJ_VER,NAMESPACE_OBJ_ID,NAMESPACE_OBJ_VER, OBJECT_VERSION_NUMBER.

11.2 Create and Modify Report Sets

This is a public interface for all functions related to Report Sets.

This section contains the following topics:

- Section 11.2.1, "Create a Report Set"
11.2.1 Create a Report Set

Use this API to create a new Report Set definition only, a new instance of an existing Report Set definition, or a new definition and an instance of it.

**Name**  
CDR_PUB_RS_REPORT_SET.CreateReportSet

**Signature**

PROCEDURE CREATEREPORTSET(
  P_API_VERSION IN NUMBER,
  P_INIT_MSG_LIST IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_COMMIT IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_VALIDATION_LEVEL IN NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
  X_RETURN_STATUS OUT VARCHAR2,
  X_MSG_COUNT OUT NUMBER,
  X_MSG_DATA OUT VARCHAR2,
  PIO_SOURCECDRNAMING IN OUT CDR_NAMING_VERSION_OBJ_TYPE,
  PI_REPORTSET IN CDR_REPORT_SET_OBJ_TYPE,
  PI_CREATEOBJECT IN VARCHAR2,
  PI_INSTANCE_SUBTYPE_ID IN CDR_NAMINGS.OBJECT_SUBTYPE_ID%TYPE,
  PI_DEFCLASSIFICATIONCOLL IN CDR_CLASSIFICATIONS_COLL,
  PI_INSTCLASSIFICATIONCOLL IN CDR_CLASSIFICATIONS_COLL
);

**Parameters**  
This API has standard parameters (see "Standard Parameters" on page 5) and the following parameters:

- **PIO_SOURCECDRNAMING** (Mandatory) This is a parameter of table type CDR_NAMING_VERSION_OBJ_TYPE that contains object attributes. Enter values for the new Report Set Entry. Use the NAMESPACE attributes to identify the Report Set or Report Set Entry in which you want to create the new Report Set Entry. For OBJECT_TYPE_RC enter $OBJTYPES$REPORTSETENTRY.

  The following attributes are required: COMPANY_ID, NAME, NAMESPACE_OBJ_ID, NAMESPACE_OBJ_VER.

- **PI_REPORTSET** (Mandatory) This is a parameter of table type CDR_REPORT_SET_OBJ_TYPE that contains object attributes.

  Enter values for the Report Set definition you want to create.

  If you are creating an instance of an existing definition, enter values to identify the definition you want to create an instance of. The following attributes are required: UNIQUE_NUMBERING_FLAG_RC, STRICT_NUMBERING_FLAG_RC.

- **PI_CREATEOBJECT** (Mandatory) Enter DEFN to create a definition only; INST to create a instance of an existing definition; or BOTH to create a new definition and an instance of it.
Create and Modify Report Sets

- **PI_INSTANCE_SUBTYPE_ID** (Optional) If you are creating a new instance, enter the ID for the subtype you want to give the instance.
  
  If you are creating a definition only, do not enter a value for this parameter.

- **PI_DEFINCLASSIFICATIONCOLL** (Optional) By default the new definition is classified according to the subtype you assigned it in the CDR_NAMING_VERSION_OBJ_TYPE.

  If you want to override the default classifications for one or more classification levels, use this parameter. This is a collection of CDR_CLA_OBJ_TYPES, which have 5 attributes, including CLA_LEVEL_ID and CLASSIFICATION_ID.

  If you want the definition to inherit its classifications for a particular level from its parent, enter the classification level ID and, for the CLASSIFICATION_ID, enter 0 (zero).

  If you want to explicitly assign one or more terms for a particular level, initialize a CDR_CLA_OBJ_TYPE for each term, entering the classification level ID and, for the CLASSIFICATION_ID, the term ID. The PAR_ attributes are not relevant to Report Sets. Do not enter any values for them. If you are not creating a new definition, do not enter values here.

- **PI_INSTCLASSIFICATIONCOLL** (Optional) By default the new instance is classified according to the subtype you assigned it in the PI_INSTANCE_SUBTYPE_ID.

  If you want to override the default classifications for one or more classification levels, use this parameter. This is a collection of CDR_CLA_OBJ_TYPES, which have 5 attributes, including CLA_LEVEL_ID and CLASSIFICATION_ID.

  If you want the instance to inherit its classifications for a particular level from its parent Work Area, enter the classification level ID and, for the CLASSIFICATION_ID, enter 0 (zero).

  If you want to explicitly assign one or more terms for a particular level, initialize a CDR_CLA_OBJ_TYPE for each term, entering the classification level ID and, for the CLASSIFICATION_ID, the term ID. The PAR_attributes are not relevant to Report Sets. Do not enter any values for them. If you are not creating a new instance, do not enter values here.

### 11.2.2 Check Out a Report Set

Use this API to check out a Report Set definition.

**Name**  
CDR_PUB_RS_REPORT_SET.CheckOutReportSet

**Signature**

PROCEDURE CHECKOUTREPORTSET(  
P_API_VERSION IN NUMBER,  
P_INIT_MSG_LIST IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,  
P_COMMIT IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,  
P_VALIDATION_LEVEL IN NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,  
X_RETURN_STATUS OUT VARCHAR2,  
X_MSG_COUNT OUT NUMBER,  
X_MSG_DATA OUT VARCHAR2,  
PIO_CDRREPORTSET IN OUT CDR_BASE_OBJ_TYPE,  
PI_COMMENT IN VARCHAR2,  
PI_ISINSTONLY IN VARCHAR2,  
PI_OPTYPE IN VARCHAR2
11.2.3 Undo a Report Set Checkout

Use this API to undo the checkout of a Report Set definition, discarding any changes that have been made.

Name  CDR_PUB_RS_REPORT_SET.UncheckOutReportSetDef

Signature

PROCEDURE UNCHECKOUTREPORTSETDEF(
  P_API_VERSION  IN    NUMBER,
  P_INIT_MSG_LIST  IN    VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_COMMIT  IN    VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_VALIDATION_LEVEL  IN    NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
  X_RETURN_STATUS  OUT    VARCHAR2,
  X_MSG_COUNT  OUT    NUMBER,
  X_MSG_DATA  OUT    VARCHAR2,
  PIO_CDRREPORTSET  IN OUT    CDR_BASE_OBJ_TYPE
);

Parameters  This API has standard parameters (see “Standard Parameters” on page 5) and the following parameter:

PIO_CDRREPORTSET (Mandatory) This is a collection of CDR_BASE_OBJTYPES. For each Report Set that you want to remove, initialize a CDR_BASE_OBJ_TYPE and then extend the collection.

The following attributes are required: COMPANY_ID,OBJ_ID,OBJ_VER,OBJECT_VERSION_NUMBER,NAMESPACE_OBJ_ID,NAMESPACE_OBJ_VER.

11.2.4 Copy Objects Into a Report Set

Use this API to copy one or more Report Set Entries and Program instances into a Report Set definition. You may specify a Report Set instance into which you want to copy the Report Set Entries. In that case, the system copies the Report Set Entries into the Report Set definition that the Report Set instance references and the Report Set instance you specify is upgraded to point to the new version of the Report Set definition.

Name  CDR_PUB_RS_REPORT_SET.CopyObjectsIntoReportSet
create and modify report sets

signature

procedure copyobjectsintoreportset(
    p_api_version in number,
    p_init_msg_list in varchar2 := cdr_pub_def_consts.g_false,
    p_commit in varchar2 := cdr_pub_def_consts.g_false,
    p_validation_level in number := cdr_pub_def_consts.g_valid_level_full,
    x_return_status out varchar2,
    x_msg_count out number,
    x_msg_data out varchar2,
    pi_cdrtargetcontainerobject in out cdr_base_obj_type,
    pi_checkinflag in varchar2,
    pi_copyprgassgnmt in varchar2
);

parameters

this api has standard parameters (see "standard parameters" on page 5) and the following parameters:

- pi_cdrbaseobjectcoll (mandatory) this is a collection of cdr_base_obj_type that is used to describe the report set entries you want to copy.
  enter values for attributes company_id, obj_id, obj_ver, namespace_obj_id, namespace_obj_ver, and object_version_number for each rse you want to copy. all children, grandchildren, etc. rses of the rses you specify are included in the copy operation.
  enter company_id, obj_id and obj_ver, namespace_obj_id, namespace_obj_ver, object_version_number for each rse

- pi_cdrtargetcontainerobject (mandatory) this is a parameter of table type cdr_base_obj_type that contains object attributes. enter values to identify the report set into which you want to copy the specified report set entries.
  the following attributes are required: company_id, obj_id and obj_ver, namespace_obj_id, namespace_obj_ver, object_version_number of target report set definition.

- pi_checkinflag (mandatory) enter 'y' so that the copied report set entries are appropriately numbered in their new location.

- pi_copyprgassgnmt (mandatory) enter 'y' to copy any program instances currently assigned to the report set entries to be copied, with their planned output assignments and mappings.
  enter 'n' to avoid copying these program instances. in this case, all planned output assignments are lost.

11.2.5 get a summary output validation status

use this api to calculate the summary output validation status for the entire report set hierarchy in the context of a report set instance.

name cdr_pub_rs_report_set.getsovsforrshierarchy

signature

function getsovsforrshierarchy(
    p_api_version in number,
    p_init_msg_list in varchar2 := cdr_pub_def_consts.g_false,
    p_commit in varchar2 := cdr_pub_def_consts.g_false,
);
Create and Modify Report Sets

Report Sets

11.2.6 Modify a Report Set

Use this API to modify a Report Set definition or instance. You can modify the name and description. If you are modifying an instance object, you can also change the 3 REF attribute values to select a different source definition.

Name  CDR_PUB_RS_REPORT_SET.ModifyReportSetDetails

Signature

PROCEDURE MODIFYREPORTSETDETAILS(
    P_API_VERSION IN NUMBER,
    P_INIT_MSG_LIST IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_COMMIT IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_VALIDATION_LEVEL IN NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
    X_RETURN_STATUS OUT VARCHAR2,
    X_MSG_COUNT OUT NUMBER,
    X_MSG_DATA OUT VARCHAR2,
    PIO_CDRNAMING IN OUT CDR_NAMING_VERSION_OBJ_TYPE,
    PI_REPORTSET IN CDR_REPORT_SET_OBJ_TYPE
) ;

Parameters  This API has standard parameters (see “Standard Parameters” on page 5) and the following parameters:

- **PIO_CDRNAMING** (Mandatory) This is a parameter of table type CDR_NAMING_VERSION_OBJ_TYPE that contains object attributes.

  Enter values to identify the Report Set and enter new values for the attributes you want to modify. You can change the name or description for either a definition or instance.
For an instance, you can also change to a different underlying source definition by entering values for the new definition in the three REF attributes. All attributes are required.

---

**Note:** Use separate APIs for modifying the validation status and the version label: CDR_PUB_VL_VALIDATION.UPDATE_VAL_STATUS and CDR_PUB_DF_NAMING.UPDATEVERSIONLABEL.

---

- **PI_REPORTSET** (Mandatory) This is a parameter of table type CDR_REPORT_SET_OBJ_TYPE that contains Report Set attributes.

  Enter values to identify the Report Set definition or instance that you want to modify and enter new values for the attributes you want to modify. You can change the TITLE, the UNIQUE_NUMBERING_FLAG_RC, and the STRICT_NUMBERING_FLAG_RC. All attributes are required.

11.2.7 Move Objects into a Report Set

Use this API to move Program instances and Report Set Entries into a Report Set definition.

**Name**  CDR_PUB_RS_REPORT_SET.MoveObjectsIntoReportSet

**Signature**

```sql
PROCEDURE MOVEOBJECTSINTOREPORTSET(
    P_API_VERSION IN    NUMBER,
    P_INIT_MSG_LIST  IN    VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_COMMIT  IN    VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_VALIDATION_LEVEL  IN    NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
    X_RETURN_STATUS  OUT    VARCHAR2,
    X_MSG_COUNT  OUT    NUMBER,
    X_MSG_DATA  OUT    VARCHAR2,
    PI_CDRBASEOBJECTCOLL  IN OUT    CDR_BASE_OBJ_COLL,
    PI_CDRTARGETCONTAINEROBJECT  IN OUT    CDR_BASE_OBJ_TYPE
);
```

**Parameters**  This API has standard parameters (see "Standard Parameters" on page 5) and the following parameters:

- **PI_CDRBASEOBJECTCOLL** (Mandatory) This is a collection of cdr_base_obj_type that is used to describe the Report Set Entries you want to copy.

  Enter values for attributes COMPANY_ID, OBJ_ID, OBJ_VER, NAMESPACE_OBJ_ID, NAMESPACE_OBJ_VER, AND OBJECT_VERSION_NUMBER for each RSE you want to copy. All children, grandchildren, etc. RSEs of the RSEs you specify are included in the Copy operation.

  Enter COMPANY_ID, OBJ_ID AND OBJ_VER, NAMESPACE_OBJ_ID, NAMESPACE_OBJ_VER, OBJECT_VERSION_NUMBER for each RSE

- **PI_CDRTARGETCONTAINEROBJECT** (Mandatory) This is a parameter of table type CDR_BASE_OBJ_TYPE that contains object attributes. Enter values to identify the Report Set into which you want to copy the specified Report Set Entries.
The following attributes are required: COMPANY_ID, OBJ_ID, OBJ_VER, NAMESPACE_OBJ_ID, NAMESPACE_OBJ_VER, OBJECT_VERSION_NUMBER of the target Report Set Definition.

### 11.2.8 Remove Objects from a Report Set

Use this API to remove one or more objects from a Report Set.

**Name**  
CDR_PUB_RS_REPORT_SET.RemoveObjectsFromReportSet

**Signature**

```sql
PROCEDURE REMOVEOBJECTSFROMREPORTSET(
  P_API_VERSION IN NUMBER,
  P_INIT_MSG_LIST IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_COMMIT IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_VALIDATION_LEVEL IN NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
  X_RETURN_STATUS OUT VARCHAR2,
  X_MSG_COUNT OUT NUMBER,
  X_MSG_DATA OUT VARCHAR2,
  PI_CDRBASEOBJECTCOLL IN OUT CDR_BASE_OBJ_COLL
);
```

**Parameters**  
This API has standard parameters (see “Standard Parameters” on page 5) and the following parameter:

- **PI_CDRBASEOBJECTCOLL** (Mandatory) This is a collection of cdr_base_obj_type that is used to describe the Report Set Entries you want to copy.

Enter values for attributes COMPANY_ID, OBJ_ID, OBJ_VER, NAMESPACE_OBJ_ID, NAMESPACE_OBJ_VER, AND OBJECT_VERSION_NUMBER for each RSE you want to copy. All children, grandchildren, etc. RSEs of the RSEs you specify are included in the Copy operation.

### 11.2.9 Check In a Report Set

Use this API to check in a Report Set definition.

**Name**  
CDR_PUB_RS_REPORT_SET.CheckInReportSetDef

**Signature**

```sql
PROCEDURE CHECKINREPORTSETDEF(
  P_API_VERSION IN NUMBER,
  P_INIT_MSG_LIST IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_COMMIT IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_VALIDATION_LEVEL IN NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
  X_RETURN_STATUS OUT VARCHAR2,
  X_MSG_COUNT OUT NUMBER,
  X_MSG_DATA OUT VARCHAR2,
  PI_CDRREPORTSET IN OUT CDR_BASE_OBJ_TYPE,
  PI_COMMENT IN VARCHAR2
);
```

**Parameters**  
This API has standard parameters (see “Standard Parameters” on page 5) and the following parameters:

- **PI_CDRREPORTSET** (Mandatory) This is a parameter of table type CDR_BASE_OBJ_TYPE that contains object attributes. Enter values to identify the Report Set definition that you want to check in.
The following attributes are required: COMPANY_ID, OBJ_ID, OBJ_VER, OBJECT_VERSION_NUMBER, NAMESPACE_OBJ_ID, NAMESPACE_OBJ_VER.

- **PI_COMMENT** (Optional) Enter the reason you are checking in the Report Set definition.

### 11.2.10 Remove a Report Set Definition

Use this API to remove a Report Set definition or instance.

**Name**  
CDR_PUB_RS_REPORT_SET.RemoveReportSet

**Signature**

```sql
PROCEDURE REMOVEREPORTSET(
    P_API_VERSION IN NUMBER,
    P_INIT_MSG_LIST IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_COMMIT IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_VALIDATION_LEVEL IN NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
    X_RETURN_STATUS OUT VARCHAR2,
    X_MSG_COUNT OUT NUMBER,
    X_MSG_DATA OUT VARCHAR2,
    PIO_CDRREPORTSET IN OUT CDR_BASE_OBJ_TYPE
);
```

**Parameters**  
This API has standard parameters (see “Standard Parameters” on page 5) and the following parameters:

**PIO_CDRREPORTSET** (Mandatory) This is a collection of CDR_BASE_OBJ_TYPES. For each Report Set that you want to remove, initialize a CDR_BASE_OBJ_TYPE and then extend the collection.

The following attributes are required: COMPANY_ID, OBJ_ID, OBJ_VER, OBJECT_VERSION_NUMBER, NAMESPACE_OBJ_ID, NAMESPACE_OBJ_VER.

### 11.2.11 Remove a Report Set

Use this API to remove one or more objects from a Report Set.

**Name**  
CDR_PUB_RS_REPORT_SET.RemoveReportSets

**Signature**

```sql
PROCEDURE REMOVEREPORTSETS(
    P_API_VERSION IN NUMBER,
    P_INIT_MSG_LIST IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_COMMIT IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_VALIDATION_LEVEL IN NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
    X_RETURN_STATUS OUT VARCHAR2,
    X_MSG_COUNT OUT NUMBER,
    X_MSG_DATA OUT VARCHAR2,
    PIO_CDRBASEOBJCOLL IN OUT CDR_BASE_OBJ_COLL
);
```

**Parameters**  
This API has standard parameters (see “Standard Parameters” on page 5) and the following parameter:

**PI_CDRBASEOBJECTCOLL** (Mandatory) This is a collection of cdr_base_obj_type that is used to describe the Report Set Entries you want to copy.
Enter values for attributes COMPANY_ID, OBJ_ID, OBJ_VER, NAMESPACE_OBJ_ID, NAMESPACE_OBJ_VER, AND OBJECT_VERSION_NUMBER for each RSE you want to copy. All children, grandchildren, etc. RSEs of the RSEs you specify are included in the Copy operation.

11.3 Create and Modify Overlay Template Definitions

This is a public interface used to create, modify and remove OTD files, which are contained by Overlay Template Definitions (OTDs) and used in Report Sets.

This section contains the following topics:

- Section 11.3.1, "Create an Overlay Template Definition"
- Section 11.3.2, "Modify an Overlay Template Definition File Definition"
- Section 11.3.3, "Get an Overlay Template Definition File as a BLOB"
- Section 11.3.4, "Remove an Overlay Template Definition File Definition"

11.3.1 Create an Overlay Template Definition

Use this API to create an OTD File definition after you have created an Overlay Template Definition (OTD). Use CDR_PUB_RS_OVERLAY_TEMPLATE.CopyObjectsIntoOTD to assign this OTD File definition to an OTD.

Name  CDR_PUB_RS_OTD_FILE.CreateOTDFile

Signature

PROCEDURE CREATEOTDFILE(  
  P_API_VERSION  IN    NUMBER,  
  P_INIT_MSG_LIST  IN    VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,  
  P_COMMIT  IN    VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,  
  P_VALIDATION_LEVEL  IN    NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,  
  X_RETURN_STATUS  OUT    VARCHAR2,  
  X_MSG_COUNT  OUT    NUMBER,  
  X_MSG_DATA  OUT    VARCHAR2,  
  PIO_OTDF_SOURCECDRNAMING  IN OUT    CDR_NAMING_VERSION_OBJ_TYPE,  
  PI_CDROTDFILEOBJTYPE  IN OUT    CDR_OTDF_OBJ_TYPE,  
  PI_LOBMODE  IN    VARCHAR := NULL
);  

Parameters  This API has standard parameters (see "Standard Parameters" on page 5) and the following parameters:

- **PIO_OTDF_SOURCECDRNAMING** (Mandatory) This is a parameter of table type CDR_NAMING_VERSION_OBJC_TYPE that contains object attributes. Enter values for the OTD File that you are creating, using the Namespace attributes to identify the parent Overlay Template Definition.

  For OBJECT_TYPE_RC enter $OBJTYPES$OVERLAYTEMPLATEFILE. The following attributes are required: COMPANY_ID,NAME,NAMESPACE_OBJ_ID,NAMESPACE_OBJ_VER.

- **PI_CDROTDFILEOBJTYPE** (Mandatory) This is a parameter of table type CDR_OTDF_OBJC_TYPE that contains object attributes specific to Overlay Template Definition Files. Enter values for the OTD File that you are creating.
The following attributes are required: OTDF_TYPE_RC, OTDF_PGNO_FLAG, OTDF_ORIENTATION_RC, OTDF_PAPERSIZE, OTDF_LANGUAGE_RC, OTDF_ROTATION, OTDF_FILENAME, OTDF_BLOB.

- **PI_LOBMODE** (Optional) Enter 'IN_DIRECT'. No other value is supported except Null.

### 11.3.2 Modify an Overlay Template Definition File Definition

Use this API to modify an OTD File definition.

**Name**  
CDR_PUB_RS_OTD_FILE.ModifyOTDFile

**Signature**

```sql
PROCEDURE MODIFYOTDFILE(
    P_API_VERSION IN NUMBER,
    P_INIT_MSG_LIST IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_COMMIT IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_VALIDATION_LEVEL IN NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
    X_RETURN_STATUS OUT VARCHAR2,
    X_MSG_COUNT OUT NUMBER,
    X_MSG_DATA OUT VARCHAR2,
    PIO_OTDF_SOURCECDRNAMING IN OUT CDR_NAMING_VERSION_OBJ_TYPE,
    PI_CDROTDFILEOBJTYPE IN OUT CDR_OTDF_OBJ_TYPE,
    PI_LOBMODE IN VARCHAR := NULL);
```

**Parameters**  
This API has standard parameters (see "Standard Parameters" on page 5) and the following parameters:

- **PIO_OTDF_SOURCECDRNAMING** (Mandatory) This is a parameter of table type CDR_NAMING_VERSION_OBJ_TYPE that contains object attributes. Enter values to identify the OTD File that you are modifying and enter new values for attributes you want to change. You can change the name and description. All attributes are required.

- **PI_CDROTDFILEOBJTYPE** (Mandatory) This is a parameter of table type CDR_OTDF_OBJ_TYPE that contains object attributes specific to Overlay Template Definition Files. Enter values to identify the OTD File that you are modifying and enter new values for attributes you want to change. You can change values for Page Number, Orientation, Paper Size, Rotation, or File Name. All attributes are required.

- **PI_LOBMODE** (Optional) Enter 'IN_DIRECT'. No other value is supported except Null.

### 11.3.3 Get an Overlay Template Definition File as a BLOB

Use this API to upload the RTF file.

**Name**  
CDR_PUB_RS_OTD_FILE.GetOTDBlob

**Signature**

```sql
FUNCTION GETOTDBLOB(
    P_API_VERSION IN NUMBER,
    P_INIT_MSG_LIST IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_COMMIT IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_VALIDATION_LEVEL IN NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
```

---

11-30  
Oracle Life Sciences Data Hub Application Programming Interface Guide
COMPANYID IN CDR_NAMINGS.COMPANY_ID%TYPE,
OBJID IN CDR_NAMINGS.OBJ_ID%TYPE,
OBJVER IN CDR_NAMING_VERSIONS.OBJ_VER%TYPE
) RETURN BLOB;

Return Type BLOB

Parameters This API has standard parameters (see "Standard Parameters" on page 5) and the following parameters:

- COMPANYID (Mandatory) Enter the Company ID of the OTD File definition.
- OBJID (Mandatory) Enter the Object ID of the OTD File definition.
- OBJVER (Mandatory) Enter the Object version of the OTD File definition.

11.3.4 Remove an Overlay Template Definition File Definition

Use this API to remove one or more OTD File definitions from an OTD.

Name CDR_PUB_RS_OTD_FILE.RemoveOTDFile

Signature

PROCEDURE REMOVEOTDFILE(
  P_API_VERSION IN NUMBER,
  P_INIT_MSG_LIST IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_COMMIT IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_VALIDATION_LEVEL IN NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
  X_RETURN_STATUS OUT VARCHAR2,
  X_MSG_COUNT OUT NUMBER,
  X_MSG_DATA OUT VARCHAR2,
  PIO_CDRBASEOBJCOLL IN OUT CDR_BASE_OBJ_COLL
);

Parameters This API has standard parameters (see "Standard Parameters" on page 5) and the following parameter:

PIO_CDRBASEOBJCOLL (Mandatory) This is a collection of CDR_BASE_OBJ_TYPEs. For each OTD File definition that you want to remove, initialize a CDR_BASE_OBJ_TYPE and then extend the collection.

The following attributes are required: COMPANY_ID,OBJ_ID,OBJ_VER,OBJECT_VERSION_NUMBER,NAMESPACE_OBJ_ID,NAMESPACE_OBJ_VER.

11.4 Report Set Overlay Template

This package is used to create, check in, check out, and modify the Report Set Entry templates.

This section contains the following topics:

- Section 11.4.1, "Create an Overlay Template"
- Section 11.4.2, "Check Out an Overlay Template"
- Section 11.4.3, "Undo an Overlay Template Checkout"
- Section 11.4.4, "Copy Objects Into an Overlay Template"
- Section 11.4.5, "Modify an Overlay Template"
- Section 11.4.6, "Check In an Overlay Template"
11.4.1 Create an Overlay Template

Use this API to create an Overlay Template definition (OTD).

**Name**  
CDR_PUB_RS_OVERLAY_TEMPLATE.CreateOverlayTemplate

**Signature**

```
PROCEDURE CREATEOVERLAYTEMPLATE(
  P_API_VERSION IN NUMBER,
  P_INIT_MSG_LIST IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_COMMIT IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_VALIDATION_LEVEL IN NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
  X_RETURN_STATUS OUT VARCHAR2,
  X_MSG_COUNT OUT NUMBER,
  X_MSG_DATA OUT VARCHAR2,
  PIO_OTD_SOURCECDRNAMING IN OUT CDR_NAMING_VERSION_OBJ_TYPE,
  PI_CDROTDOBJTYPE IN OUT CDR_OTD_OBJ_TYPE,
  PO_DEFCLASSIFICATIONCOLL IN CDR_CLASSIFICATIONS_COLL
);
```

**Parameters**  
This API has standard parameters (see "Standard Parameters" on page 5) and the following parameters:

- **PIO_OTD_SOURCECDRNAMING** (Mandatory) This is a parameter of table type CDR_NAMING_VERSION_OBJ_TYPE that contains object attributes. Enter values for the Overlay Template Definition (OTD) that you are creating.

  For OBJECT_TYPE_RC enter $OBJTYPES$OVERLAYTEMPLATE. The following attributes are required: COMPANY_ID, NAME, NAMESPACE_OBJ_ID, NAMESPACE_OBJ_VER.

- **PI_CDROTDOBJTYPE** (Mandatory) This is a parameter of table type CDR_OTD_OBJ_TYPE that contains object attributes specific to Overlay Templates. Enter values for the Overlay Template that you are creating.

  The following attributes are required: OTD_DEFAULT_PAPERSIZE, OTD_DEFAULT_LANGUAGE_RC.

- **PO_DEFCLASSIFICATIONCOLL** (Optional) By default the new definition is classified according to the subtype you assigned it in the CDR_NAMING_VERSION_OBJ_TYPE.

  If you want to override the default classifications for one or more classification levels, use this parameter. This is a collection of CDR_CLA_OBJ_TYPES, which have 5 attributes, including CLA_LEVEL_ID and CLASSIFICATION_ID.

  If you want the definition to inherit its classifications for a particular level from its parent, enter the classification level ID and, for the CLASSIFICATION_ID, enter 0 (zero).

  If you want to explicitly assign one or more terms for a particular level, initialize a CDR_CLA_OBJ_TYPE for each term, entering the classification level ID and, for the CLASSIFICATION_ID, the term ID. The PAR_ attributes are not relevant to OTDs. Do not enter any values for them.

11.4.2 Check Out an Overlay Template

Use this API to check out an Overlay Template Definition (OTD).
Name  CDR_PUB_RS_OVERLAY_TEMPLATE.CheckOutOverlayTemplate

Signature

PROCEDURE CHECKOUTOVERLAYTEMPLATE(
  P_API_VERSION IN    NUMBER,
  P_INIT_MSG_LIST IN    VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_COMMIT IN    VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_VALIDATION_LEVEL IN    NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
  X_RETURN_STATUS OUT    VARCHAR2,
  X_MSG_COUNT OUT    NUMBER,
  X_MSG_DATA OUT    VARCHAR2,
  PIO_BASEOBJECT IN OUT    CDR_BASE_OBJ_TYPE,
  PI_COMMENT IN    VARCHAR2,
  PI_ISINSTONLY IN    VARCHAR2
);

Parameters  This API has standard parameters (see “Standard Parameters” on page 5) and the following parameters:

- **PIO_BASEOBJECT** (Mandatory) This is a parameter of table type CDR_BASE_OBJ_TYPE that contains object attributes. Enter values to identify the OTD that you want to check in.
  
  The following attributes are required: COMPANY_ID,OBJ_ID,OBJ_VER,OBJECT_VERSION_NUMBER,NAMESPACE_OBJ_ID,NAMESPACE_OBJ_VER.

- **PI_COMMENT** (Optional) Enter the reason you are checking in the OTD.

- **PI_ISINSTONLY** (Mandatory) Enter $YESNO$NO.

11.4.3 Undo an Overlay Template Checkout

Use this API to undo the checkout of an Overlay Template Definition (OTD).

Name  CDR_PUB_RS_OVERLAY_TEMPLATE.UndoCheckOutOverlayTemplate

Signature

PROCEDURE UNDOCHECKOUTOVERLAYTEMPLATE(
  P_API_VERSION IN    NUMBER,
  P_INIT_MSG_LIST IN    VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_COMMIT IN    VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_VALIDATION_LEVEL IN    NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
  X_RETURN_STATUS OUT    VARCHAR2,
  X_MSG_COUNT OUT    NUMBER,
  X_MSG_DATA OUT    VARCHAR2,
  PIO_BASEOBJECT IN OUT    CDR_BASE_OBJ_TYPE
);

Parameters  This API has standard parameters (see “Standard Parameters” on page 5) and the following parameters:

- **PIO_BASEOBJECT** (Mandatory) This is a parameter of table type CDR_BASE_OBJ_TYPE that contains object attributes. Enter values to identify the OTD whose checkout you want to undo.
  
  The following attributes are required: COMPANY_ID,OBJ_ID,OBJ_VER,OBJECT_VERSION_NUMBER,NAMESPACE_OBJ_ID,NAMESPACE_OBJ_VER.
11.4.4 Copy Objects Into an Overlay Template

Use this API to copy OTD File definitions into an Overlay Template Definition.

**Name**  CDR_PUB_RS_OVERLAY_TEMPLATE.CopyObjectsIntoOTD

**Signature**

PROCEDURE COPYOBJECTSINTOOTD(
  P_API_VERSION  IN    NUMBER,
  P_INIT_MSG_LIST  IN    VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_COMMIT  IN    VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_VALIDATION_LEVEL  IN    NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
  X_RETURN_STATUS  OUT    VARCHAR2,
  X_MSG_COUNT  OUT    NUMBER,
  X_MSG_DATA  OUT    VARCHAR2,
  PI_CDRBASEOBJCOLL  IN    CDR_BASE_OBJ_COLL,
  PI_CDRTARGETCONTAINEROBJECT  IN    CDR_BASE_OBJ_TYPE
);

**Parameters**  This API has standard parameters (see "Standard Parameters" on page 5) and the following parameters:

- **PI_CDRBASEOBJCOLL** (Mandatory) This is a collection of CDR_BASE_OBJ_TYPES. For each OTD file definition that you want to copy into an OTD, initialize a CDR_BASE_OBJ_TYPE and then extend the collection.

  The following attributes are required: COMPANY_ID,OBJ_ID,OBJ_VER,OBJECT_VERSION_NUMBER,NAMESPACE_OBJ_ID,NAMESPACE_OBJ_VER.

- **PI_CDRTARGETCONTAINEROBJECT** (Mandatory) This is a parameter of table type CDR_BASE_OBJ_TYPE that contains object attributes. Enter values to identify the OTD into which you want to copy OTD File definitions. The following attributes are required: COMPANY_ID,OBJ_ID,OBJ_VER,OBJECT_VERSION_NUMBER,NAMESPACE_OBJ_ID,NAMESPACE_OBJ_VER.

11.4.5 Modify an Overlay Template

Use this API to modify an Overlay Template Definition (OTD).

**Name**  CDR_PUB_RS_OVERLAY_TEMPLATE.ModifyOverlayTemplate

**Signature**

PROCEDURE MODIFYOVERLAYTEMPLATE(
  P_API_VERSION  IN    NUMBER,
  P_INIT_MSG_LIST  IN    VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_COMMIT  IN    VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_VALIDATION_LEVEL  IN    NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
  X_RETURN_STATUS  OUT    VARCHAR2,
  X_MSG_COUNT  OUT    NUMBER,
  X_MSG_DATA  OUT    VARCHAR2,
  PIO_OTD_SOURCECDRNAMING  IN OUT    CDR_NAMING_VERSION_OBJ_TYPE,
  PI_CDROTDOBJTYPE  IN OUT    CDR_OTD_OBJ_TYPE
);

**Parameters**  This API has standard parameters (see "Standard Parameters" on page 5) and the following parameters:

- **PIO_OTD_SOURCECDRNAMING** (Mandatory) This is a parameter of table type CDR_NAMING_VERSION_OBJ_TYPE that contains object attributes. Enter
values to identify the Overlay Template Definition (OTD) that you are modifying, and enter new values for the attributes you want to modify. You can change the name and description. All attributes are required.

- **PI_CDROTDOBJTYPE** Object Type of Overlay Template Definition

### 11.4.6 Check In an Overlay Template

Use this API to check in an Overlay Template Definition.

**Name**  CDR_PUB_RS_OVERLAY_TEMPLATE.CheckInOverlayTemplate

**Signature**

```sql
PROCEDURE CHECKINOVERLAYTEMPLATE(
    P_API_VERSION IN NUMBER,
    P_INIT_MSG_LIST IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_COMMIT IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_VALIDATION_LEVEL IN NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
    X_RETURN_STATUS OUT VARCHAR2,
    X_MSG_COUNT OUT NUMBER,
    X_MSG_DATA OUT VARCHAR2,
    PIO_BASEOBJECT IN OUT CDR_BASE_OBJ_TYPE,
    PI_COMMENT IN VARCHAR2 );
```

**Parameters**  This API has standard parameters (see "Standard Parameters" on page 5) and the following parameters:

- **PIO_BASEOBJECT** (Mandatory) This is a parameter of table type CDR_BASE_OBJ_TYPE that contains object attributes. Enter values to identify the OTD that you want to check in.
  
  The following attributes are required: COMPANY_ID, OBJ_ID, OBJ_VER, OBJECT_VERSION_NUMBER, NAMESPACE_OBJ_ID, NAMESPACE_OBJ_VER.

- **PI_COMMENT** (Optional) Enter the reason you are checking in the OTD.

### 11.4.7 Remove an Overlay Template

Use this API to remove one or more Overlay Template Definitions (OTDs).

**Name**  CDR_PUB_RS_OVERLAY_TEMPLATE.RemoveOverlayTemplate

**Signature**

```sql
PROCEDURE REMOVEOVERLAYTEMPLATE(
    P_API_VERSION IN NUMBER,
    P_INIT_MSG_LIST IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_COMMIT IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_VALIDATION_LEVEL IN NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
    X_RETURN_STATUS OUT VARCHAR2,
    X_MSG_COUNT OUT NUMBER,
    X_MSG_DATA OUT VARCHAR2,
    PIO_CDRBASEOBJCOLL IN OUT CDR_BASE_OBJ_COLL
    );
```

**Parameters**  This API has standard parameters (see "Standard Parameters" on page 5) and the following parameter:
PIO_CDRBASEOBJCOLL (Mandatory) This is a collection of CDR_BASE_OBJ_TYPEs. For each OTD that you want to remove, initialize a CDR_BASE_OBJ_Type and then extend the collection.

The following attributes are required: COMPANY_ID,OBJ_ID,OBJ_VER,OBJECT_VERSION_NUMBER,NAMESPACE_OBJ_ID,NAMESPACE_OBJ_VER.
This is a public interface for Source Code-related operations including creating, modifying, and removing Source Code objects.

12.1 Create and Modify Source Code

This section contains the following topics:

- Section 12.1.1, "Create a Source Code Object"
- Section 12.1.2, "Get a Source Code CLOB"
- Section 12.1.3, "Modify Source Code"
- Section 12.1.4, "Set the Primary Flag to Yes"
- Section 12.1.5, "Update a Shareable Flag"
- Section 12.1.6, "Remove a Source Code Object"

12.1.1 Create a Source Code Object

Use this API to create a new instance of an existing Source Code definition or a new Source Code definition and an instance of it.

**Note:** Source Code definitions and instances are always contained in Program definitions. You cannot create a Source Code definition without also creating an instance of it in the same Program definition.

**Name**  CDR_PUB_DF_SOURCECODE.CreateSourceCode

**Signature**

PROCEDURE CREATESOURCECODE(
  P_API_VERSION IN NUMBER,
  P_INIT_MSG_LIST IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_COMMIT IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_VALIDATION_LEVEL IN NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
  X_RETURN_STATUS OUT VARCHAR2,
  X_MSG_COUNT OUT NUMBER,
  X_MSG_DATA OUT VARCHAR2,
  PIO_SCREF_SOURCECDRNAMING IN OUT CDR_NAMING_VERSION_OBJ_TYPE,
  PI_CDRSCOBJTYPE IN OUT CDR_SRCCODE_OBJ_TYPE,
  PIO_CDRSCREFOBJTYPE IN OUT CDR_SRCCODE_REF_OBJ_TYPE,
  PI_CREATEOBJECT IN VARCHAR2,
  PI_DEFINITON_SUBTYPE_ID IN CDR_NAMINGS.OBJECT_SUBTYPE_ID%TYPE,
Create and Modify Source Code

```
PI_VLOBMODE IN VARCHAR2 := NULL,
PIO_CDRSCBLOB IN OUT CDR_SRCCODE_BLOB_OBJ_TYPE,
PIO_CDRSCCLOB IN OUT CDR_SRCCODE_CLOB_OBJ_TYPE
);
```

**Parameters**  
This API has standard parameters (see "Standard Parameters" on page 5) and the following parameters:

- **PIO_SCREF_SOURCECDRNAMING** (Mandatory) This is a parameter of table type CDR_NAMING_VERSIONS_OBJ_TYPE that contains CDR Naming Version attributes.
  
Enter values for the Source Code instance you are creating. For OBJECT_TYPE_RC enter $OBJTYPES$SRCCDEREF.

- **PI_CDRSCOBJTYPE** (Optional) This is a parameter of table type CDR_SRCCODE_OBJ_TYPE that contains object attributes specific to Source Code definitions.
  
If you are creating a new definition, enter values for the new Source Code definition. The following attributes are required: TECH_TYPE_ID,SRCCODE_TYPE_RC,SHAREABLE_FLAG_RC,ORACLE_PACKAGE_NAME.

If you are creating an instance of an existing Source Code definition, do not enter any values here.

- **PIO_CDRSCREFOBJTYPE** (Optional) This is a parameter of table type CDR_SRCCODE_REF_OBJ_TYPE that contains object attributes specific to Source Code instances.
  
If you are creating a new instance, enter values for it.

If you are creating a new Source Code definition only, do not enter any values here.

- **PI_CREATEOBJECT** (Mandatory) Enter INST to create a instance of an existing definition or BOTH to create a new definition and an instance of it.

- **PI_DEFINITON_SUBTYPE_ID** (Optional) Enter a subtype for the Source Code definition.
  
If you do not enter a value, the API creates the definition with the default subtype.

- **PI_VLOBMODE** Enter 'DIRECT' if your source code is already contained in a BLOB or CLOB file. You enter information about it in one of the next two parameters and the API uploads it immediately.
  
If you enter anything other than 'DIRECT' the API creates a new, empty BLOB and CLOB in the next parameters. It prompts you to paste your source code in and then uploads the BLOB or CLOB.

- **PIO_CDRSCBLOB** This is a compound object of type CDR_SRCCODE_BLOB_OBJ_TYPE.
  
If the source code is binary, enter its name for the FILE_NAME attribute value and the BLOB itself for the FILE_BLOB attribute value.

- **PIO_CDRSCCLOB** This is a compound object of type CDR_SRCCODE_CLOB_OBJ_TYPE.
  
If the source code is text-based, enter its name for the FILE_NAME attribute value and the CLOB itself for the FILE_CLOB attribute value.
12.1.2 Get a Source Code CLOB

Use this API to get the source code CLOB.

**Name**  CDR_PUB_DF_SOURCECODE.GetSourceCode

**Signature**

FUNCTION GETSOURCECODE(
  P_API_VERSION  IN    NUMBER,
  P_INIT_MSG_LIST  IN    VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_COMMIT  IN    VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_VALIDATION_LEVEL  IN    NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
  COMPANYID  IN    CDR_NAMINGS.COMPANY_ID%TYPE,
  OBJID  IN    CDR_NAMINGS.OBJ_ID%TYPE,
  OBJVER  IN    CDR_NAMING_VERSIONS.OBJ_VER%TYPE,
  NSOBJID  IN    CDR_NAMINGS.OBJ_ID%TYPE,
  NSOBJVER  IN    CDR_NAMING_VERSIONS.OBJ_VER%TYPE
) RETURN CLOB;

**Return**  Type CLOB

**Parameters**  This API has standard parameters (see “Standard Parameters” on page 5) and the following parameters:

- **COMPANYID** (Mandatory) Enter your company ID.
- **OBJID** (Mandatory) Enter the object ID of the Source Code definition.
- **OBJVER** (Mandatory) Enter the object version (OBJ_VER) of the Source Code definition.
- **NSOBJID** (Mandatory) Enter the object ID of the Program definition that contains the Source Code definition.
- **NSOBJVER** (Mandatory) Enter the object version (OBJ_VER) of the Program definition that contains the Source Code definition.

12.1.3 Modify Source Code

Use this API to modify a Source Code definition or instance. You can modify the name and description. If you are modifying a Source Code instance, you can also change the 3 REF attribute values to select a different source definition.

**Name**  CDR_PUB_DF_SOURCECODE.ModifySourceCodeDetails

**Signature**

PROCEDURE MODIFYSOURCECODEDETAILS(
  P_API_VERSION  IN    NUMBER,
  P_INIT_MSG_LIST  IN    VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_COMMIT  IN    VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_VALIDATION_LEVEL  IN    NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
  X_RETURN_STATUS  OUT    VARCHAR2,
  X_MSG_COUNT  OUT    NUMBER,
  X_MSG_DATA  OUT    VARCHAR2,
  PIO_SCREF_SOURCECDRNAMING  IN OUT    CDR_NAMING_VERSION_OBJ_TYPE,
  PI_CDRSCOBJTYPE  IN OUT    CDR_SRCCODE_OBJ_TYPE,
  PIO_CDRSCREFOBJTYPE  IN OUT    CDR_SRCCODE_REF_OBJ_TYPE,
  PI_VLOBMODE  IN    VARCHAR2 := NULL,
  PIO_CDRSCBLOB  IN OUT    CDR_SRCCODE_BLOB_OBJ_TYPE,
Parameters This API has standard parameters (see "Standard Parameters" on page 5) and the following parameters:

- **PIO_SCREF_SOURCECDRNAMING** (Mandatory) This is a parameter of table type CDR_NAMING_VERSION_OBJ_TYPE that contains object attributes. Enter values to identify the Source Code definition or instance and enter new values for the attributes you want to modify.

  You can change the name or description for either the definition or instance. For an instance you can also change to a different underlying source definition by entering values for the new definition in the three REF attributes.

  **Note:** Neither Source Code definitions nor instances can have a validation status or a version label. All attributes are required.

- **PI_CDRSCOBJTYPE** (Mandatory) This is a parameter of table type CDR_SRCCODE_OBJ_TYPE that contains object attributes specific to Source Code definitions.

  If you are modifying a Source Code definition, enter values to identify the definition and enter new values for the attributes you want to modify.

  You can modify SHAREABLE_FLAG_RC, ORACLE_PACKAGE_NAME, and ORACLE_PROCEDURE_NAME. All attributes are required.

- **PIO_CDRSCREFOBJTYPE** (Mandatory) This is a parameter of table type CDR_SRCCODE_REF_OBJ_TYPE that contains object attributes specific to Source Code instances.

  If you are modifying a Source Code instance, enter values to identify the instance and enter new values for the attributes you want to modify. You can modify PRIMARY_FLAG_RC and FILEREF. All attributes are required.

- **PI_VLOBMODE** Enter 'DIRECT' if your source code is already contained in a BLOB or CLOB file. You enter information about it in one of the next two parameters and the API uploads it immediately.

  If you enter anything other than 'DIRECT' the API creates a new, empty BLOB and CLOB in the next parameters. It prompts you to paste in your source code and then uploads the BLOB or CLOB.

- **PIO_CDRSCBLOB** This is a compound object of type CDR_SRCCODE_BLOB_OBJ_TYPE.

  If the source code is binary, enter its name for the FILE_NAME attribute value and the BLOB itself for the FILE_BLOB attribute value.

- **PIO_CDRSCCLOB** This is a compound object of type CDR_SRCCODE_CLOB_OBJ_TYPE.

  If the source code is text-based, enter its name for the FILE_NAME attribute value and the CLOB itself for the FILE_CLOB attribute value. Enter CLOB if the source code file is a character large object.
12.1.4 Set the Primary Flag to Yes

Use this API to set the Primary Flag attribute of a specified Source Code instance to Yes. If any other Source Code instance in the parent Program is currently set to Yes, the API resets its flag to No.

Name  CDR_PUB_DF_SOURCECODE.SetPrimaryFlagRC

Signature

PROCEDURE SETPRIMARYFLAGRC(
    P_API_VERSION  IN    NUMBER,
    P_INIT_MSG_LIST  IN    VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_COMMIT  IN    VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_VALIDATION_LEVEL  IN    NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
    X_RETURN_STATUS  OUT    VARCHAR2,
    X_MSG_COUNT  OUT    NUMBER,
    X_MSG_DATA  OUT    VARCHAR2,
    PI_CDRSCREFBASEOBJTYPE  IN    CDR_BASE_OBJ_TYPE
);

Parameters  This API has standard parameters (see “Standard Parameters” on page 5) and the following parameters:

PI_CDRSCREFBASEOBJTYPE (Mandatory) This is a parameter of table type CDR_BASE_OBJ_TYPE that contains object attributes. Enter values to identify the Source Code instance whose PRIMARY_FLAG_RC you want to set to $YESNO$YES.

The following attributes are required: COMPANY_ID,OBJ_ID,OBJ_VER,OBJECT_VERSION_NUMBER,NAMESPACE_OBJ_ID,NAMESPACE_OBJ_VER.

12.1.5 Update a Shareable Flag

Use this API to set the Shearable attribute for a Source Code definition.

Name  CDR_PUB_DF_SOURCECODE.UpdateShareableFlagRC

Signature

PROCEDURE UPDATESHAREABLEFLAGRC(
    P_API_VERSION  IN    NUMBER,
    P_INIT_MSG_LIST  IN    VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_COMMIT  IN    VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_VALIDATION_LEVEL  IN    NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
    X_RETURN_STATUS  OUT    VARCHAR2,
    X_MSG_COUNT  OUT    NUMBER,
    X_MSG_DATA  OUT    VARCHAR2,
    PI_CDRSCBASEOBJTYPE  IN OUT    CDR_BASE_OBJ_TYPE,
    PI_CHANGESTATUSTO  IN    VARCHAR2
);

Parameters  This API has standard parameters (see “Standard Parameters” on page 5) and the following parameters:

PI_CDRSCBASEOBJTYPE (Mandatory) This is a parameter of table type CDR_BASE_OBJ_TYPE that contains object attributes. Enter values to identify the Source Code definition that you want to make sharable or not sharable.

The following attributes are required: COMPANY_ID,OBJ_ID,OBJ_VER,OBJECT_VERSION_NUMBER,NAMESPACE_OBJ_ID,NAMESPACE_OBJ_VER.
- **PI_CHANGESTATUSTO** (Mandatory) Enter SET to set the SHAREABLE_FLAG_RC to 'YESNO$YES' or RESET to set the SHAREABLE_FLAG_RC to 'YESNO$NO'.

### 12.1.6 Remove a Source Code Object

Use this API to remove one or more Source Code definitions or instances.

**Name**  CDR_PUB_DF_SOURCECODE.RemoveSourceCode

**Signature**

```sql
PROCEDURE REMOVESOURCECODE(
    P_API_VERSION  IN    NUMBER,
    P_INIT_MSG_LIST  IN    VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_COMMIT  IN    VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_VALIDATION_LEVEL  IN    NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
    X_RETURN_STATUS  OUT    VARCHAR2,
    X_MSG_COUNT  OUT    NUMBER,
    X_MSG_DATA  OUT    VARCHAR2,
    PIO_CDRBASEOBJCOLL  IN OUT    CDR_BASE_OBJ_COLL
);
```

**Parameters**  This API has standard parameters (see "Standard Parameters" on page 5) and the following parameters:

**PIO_CDRBASEOBJCOLL** (Mandatory) This is a collection of CDR_BASE_OBJ_TYPEs.

For each Source Code that you want to remove, initialize a CDR_BASE_OBJ_TYPE and then extend the collection. The following attributes are required: COMPANY_ID,OBJ_ID,OBJ_VER,OBJECT_VERSION_NUMBER,NAMESPACE_OBJ_ID,NAMESPACE_OBJ_VER.
This is a public interface for all operations related to Tables, Columns, and Constraints; including creation, deletion, modification, and checking in and out of these objects.

13.1 Create and Modify Tables

This section contains the following topics:

- Section 13.1.1, "Create a Table Definition"
- Section 13.1.2, "Create a Table Instance"
- Section 13.1.3, "Create a Temporary Blob"
- Section 13.1.4, "Create a Table Column"
- Section 13.1.5, "Create a Table Constraint"
- Section 13.1.6, "Modify a Table Definition"
- Section 13.1.7, "Modify a Table Descriptor"
- Section 13.1.8, "Modify a Table Instance"
- Section 13.1.9, "Reorder a Column"
- Section 13.1.10, "Upload a Table Descriptor or Column"
- Section 13.1.11, "Check In a Table Object"
- Section 13.1.12, "Remove a Single Object"

13.1.1 Create a Table Definition

Use this API to create a Table definition.

Name CDR_PUB_DF_TABLE.CreateTableDefinition

Signature

PROCEDURE CREATETABLEDEFINITION(
  P_API_VERSION IN NUMBER,
  P_INIT_MSG_LIST IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_COMMIT IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_VALIDATION_LEVEL IN NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
  X_RETURN_STATUS OUT VARCHAR2,
  X_MSG_COUNT OUT NUMBER,
  X_MSG_DATA OUT VARCHAR2,
  PIO_NAMING IN OUT CDR_NAMING_VERSION_OBJ_TYPE,
  PIO_TABLE IN OUT CDR_TABLE_OBJ_TYPE,
Create and Modify Tables

Parameters

This API has standard parameters (see "Standard Parameters" on page 5) and the following parameters:

- **PIO_NAMING** (Mandatory) This is a parameter of table type CDR_NAMING_VERSIONS_OBJ_TYPE that contains CDR Naming Version attributes. For OBJECT_TYPE_RC enter $OBJTYPES$TABLE.

- **PIO_TABLE** (Optional) This is a parameter of table type CDR_TABLE_OBJ_TYPE that contains object attributes specific to Tables.

  If you are creating a new definition, enter values for the new Table. The following attributes are required: COMPANY_ID, OBJ_ID, OBJ_VER, ORACLE_NAME, SAS_NAME, SASV6FLAGRC = '$YESNO$NO', SNAPSHOTFLAGRC = '$YESNO$YES', BLINDINGFLAGRC = '$YESNO$YES', PROCESSTYPERC = '$PROCESSTYPES$STAGINGWAUDIT'

  If you are creating an instance of an existing Table, do not enter any values here.

- **PI_INSTANCESUBTYPEID** (Mandatory) Enter NULL here.

- **PI_DEFCLASSIFICATIONCOLL** (Optional) By default the new definition is classified according to the subtype you assigned it in the CDR_NAMING_VERSION_OBJ_TYPE.

  If you want to override the default classifications for one or more classification levels, use this parameter. This is a collection of CDR_CLA_OBJ_TYPEs, which have 5 attributes, including CLA_LEVEL_ID and CLASSIFICATION_ID.

  If you want the definition to inherit its classifications for a particular level from its parent, enter the classification level ID and, for the CLASSIFICATION_ID, enter 0 (zero).

  If you want to explicitly assign one or more terms for a particular level, initialize a CDR_CLA_OBJ_TYPE for each term, entering the classification level ID and, for the CLASSIFICATION_ID, the term ID.

  The PAR_ attributes are not relevant to Tables. Do not enter any values for them.

- **PI_INSTCLASSIFICATIONCOLL** (Mandatory) Enter NULL because you are creating a Table definition.

13.1.2 Create a Table Instance

Use this API to create a Table Instance.

**Name**

CDR_PUB_DF_TABLE.CreateTableInstance

**Signature**

PROCEDURE CREATEINSTANCE(
    P_API_VERSION IN NUMBER,
    P_INIT_MSG_LIST IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_COMMIT IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_VALIDATION_LEVEL IN NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
    X_RETURN_STATUS OUT VARCHAR2,
    X_MSG_COUNT OUT NUMBER,
    X_MSG_DATA OUT VARCHAR2,
Create and Modify Tables

PIO_NAMING IN OUT CDR_NAMING_VERSION_OBJ_TYPE,
PIO_TABLE_INSTANCE IN OUT CDR_TABLE_REF_OBJ_TYPE,
PI_CREATETYPE IN VARCHAR2,
PI_INSTANCESUBTYPEID IN NUMBER,
PI_DEFCLASSIFICATIONCOLL IN CDR_CLASSIFICATIONS_COLL,
PI_INSTCLASSIFICATIONCOLL IN CDR_CLASSIFICATIONS_COLL
);

Parameters

This API has standard parameters (see "Standard Parameters" on page 5) and the following parameters:

- **PIO_NAMING** (Mandatory) This is a parameter of table type CDR_NAMING_VERSIONS_OBJ_TYPE that contains CDR Naming Version attributes. For OBJECT_TYPE_RC enter $OBJTYPES$TABLEREF.

- **PIO_TABLE_INSTANCE** (Optional) This is a parameter of table type CDR_TABLE_OBJ_TYPE that contains object attributes specific to Table instances.

  The following attributes are required: COMPANY_ID, OBJ_ID, OBJ_VER, ORACLE_NAME, SAS_NAME. SASV6FLAGRC = 'YESNO'NO', SNAPSHOTFLAGRC = 'YESNO'YES', BLINDINGFLAGRC = 'YESNO'YES', PROCESSSTYPERC = 'STAGINGWAUDIT', BLINDINGSTATUSRC = 'BLINDED', COMPRESSFLAGRC = 'YESNO'NO', GENERATIONSTATUSRC = 'YESNO'NO'.

- **PI_CREATETYPE** (Mandatory) Enter INST to create a Table instance of an existing Table definition. Enter BOTH to create a Table definition and a Table instance of it.

- **PI_INSTANCESUBTYPEID** (Mandatory) Enter the SUBTYPE_ID of the Table instance.

- **PI_DEFCLASSIFICATIONCOLL** (Optional) By default the new definition is classified according to the subtype you assigned it in the CDR_NAMING_VERSION_OBJ_TYPE.

  If you want to override the default classifications for one or more classification levels, use this parameter. This is a collection of CDR_CLA_OBJ_TYPEs, which have 5 attributes, including CLA_LEVEL_ID and CLASSIFICATION_ID.

  If you want the definition to inherit its classifications for a particular level from its parent, enter the classification level ID and, for the CLASSIFICATION_ID, enter 0 (zero).

  If you want to explicitly assign one or more terms for a particular level, initialize a CDR_CLA_OBJ_TYPE for each term, entering the classification level ID and, for the CLASSIFICATION_ID, the term ID. The PAR_attributes are not relevant to Tables. Do not enter any values for them. If you are not creating a new definition, do not enter values here.

- **PI_INSTCLASSIFICATIONCOLL** (Optional) By default the new instance is classified according to the subtype you assigned it in the PI_INSTANCE_SUBTYPE_ID.

  If you want to override the default classifications for one or more classification levels, use this parameter. This is a collection of CDR_CLA_OBJ_TYPEs, which have 5 attributes, including CLA_LEVEL_ID and CLASSIFICATION_ID.

  If you want the instance to inherit its classifications for a particular level from its parent Work Area, enter the classification level ID and, for the CLASSIFICATION_ID, enter 0 (zero).
If you want to explicitly assign one or more terms for a particular level, initialize a CDR_CLA_OBJ_TYPE for each term, entering the classification level ID and, for the CLASSIFICATION_ID, the term ID.

The PAR_ attributes are not relevant to Tables. Do not enter any values for them. If you are not creating a new instance, do not enter values here.

13.1.3 Create a Temporary Blob

Use this API to create a temporary BLOB in a BLOB Table. Call this API before you upload Columns or Table Descriptors using the UPLOADOPERATORCOLUMNS API for a SAS or CPORT file, or before creating a Table for a SAS dataset.

Name CDR_PUB_DF_TABLE.CreateTempBlob

Signature

PROCEDURE CREATETEMPBLOB {
    P_API_VERSION IN NUMBER,
    P_INIT_MSG_LIST IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_COMMIT IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_VALIDATION_LEVEL IN NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
    X_RETURN_STATUS OUT VARCHAR2,
    X_MSG_COUNT OUT NUMBER,
    X_MSG_DATA OUT VARCHAR2,
    PIO_TMPBLOBOBJ IN OUT CDR_TEMP_BLOBS_OBJ_TYPE
};

Parameters This API has standard parameters (see "Standard Parameters" on page 5) and the following parameter:

PIO_TMPBLOBOBJ (Mandatory) This is parameter of table type CDR_TEMP_BLOBS_OBJ_TYPE that contains information about the BLOB.

The required attributes are: FILE_NAME, FILE_BLOB

13.1.4 Create a Table Column

Use this API to create a Table Column. As with any other object, you can create an instance of an existing Variable or create a new Variable and Column. (Variable is the definition object for Columns.)

Name CDR_PUB_DF_TABLE.CreateColumn

Signature

PROCEDURE CREATECOLUMN {
    P_API_VERSION IN NUMBER,
    P_INIT_MSG_LIST IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_COMMIT IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_VALIDATION_LEVEL IN NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
    X_RETURN_STATUS OUT VARCHAR2,
    X_MSG_COUNT OUT NUMBER,
    X_MSG_DATA OUT VARCHAR2,
    PIO_NAMING IN OUT CDR_NAMING_VERSION_OBJ_TYPE,
    PIO_VARIABLE IN OUT CDR_VAR_OBJ_TYPE,
    PIO_COLUMN IN OUT CDR_COLUMNS_OBJ_TYPE,
    PI_CREATETYPE IN VARCHAR2,
    PI_DEFCLASSIFICATIONCOLL IN CDR_CLASSIFICATIONS_COLL
};
Parameters  This API has standard parameters (see "Standard Parameters" on page 5) and the following parameters:

- **PIO_NAMING** (Mandatory) This is a parameter of table type CDR_NAMING_VERSIONS_OBJ_TYPE that contains CDR Naming Version attributes. For OBJECT_TYPE_RC enter $OBJTYPES$COLUMN.

- **PIO_VARIABLE** (Mandatory) This is a parameter of table type CDR_VAR_OBJ_TYPE that contains object attributes specific to Variables.

  The following attributes are required: ORACLE_NAME, ORACLE_DATATYPE_RC, LENGTH, PRECISION, SAS_V8_NAME, SAS_LABEL, SAS_FORMAT, NULLABLE_FLAG_RC, DEFAULT_VALUE.

- **PIO_COLUMN** (Mandatory) This is a parameter of table type CDR_COLUMNS_OBJ_TYPE that contains object attributes specific to Columns.

  The following attributes are required: COMPANY_ID, OBJ_ID, OBJ_VER, POSITION, SAS_LABEL, NULLABLE_FLAG_RC.

- **PI_CREATETYPE** (Mandatory) Enter INST to create only a Column of an existing definition (Variable); or BOTH to create a new Column and Variable.

- **PI_DEFCLASSIFICATIONCOLL** (Optional) By default the new definition is classified according to the subtype assigned it in the CDR_NAMING_VERSION_OBJECT_TYPE.

  If you want to override the default classifications for one or more classification levels, use this parameter. This is a collection of CDR_CLA_OBJ_TYPES, which have 5 attributes, including CLA_LEVEL_ID and CLASSIFICATION_ID.

  If you want the definition to inherit its classifications for a particular level from its parent, enter the classification level ID and, for the CLASSIFICATION_ID, enter 0 (zero).

  If you want to explicitly assign one or more terms for a particular level, initialize a CDR_CLA_OBJ_TYPE for each term, entering the classification level ID and, for the CLASSIFICATION_ID, the term ID. The PAR_ attributes are not relevant to Columns. Do not enter any values for them.

  If you are not creating a new definition, do not enter values here.

13.1.5 Create a Table Constraint

Use this API to create a constraint for a Table definition or a Table instance.

**Name**  CDR_PUB_DF_TABLE.CreateTableConstraint

**Signature**

PROCEDURE CREATETABLECONSTRAINT(
    _P_API_VERSION IN NUMBER,
    _P_INIT_MSG_LIST IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    _P_COMMIT IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    _P_VALIDATION_LEVEL IN NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
    _X_RETURN_STATUS OUT VARCHAR2,
    _X_MSG_COUNT OUT NUMBER,
    _X_MSG_DATA OUT VARCHAR2,
    _PIO_NAMING IN OUT CDR_NAMING_VERSIONS_OBJ_TYPE,
    _PIO_CONSTRAINT IN OUT CDR_TABLE_CONS_OBJ_TYPE,
    _PIO_CONSTRAINTCOLUMNS IN OUT CDR_TABLE_CONCOLS_LIST_COLL,
    _PI_VALS IN CDR_VALS_COLL)
Parameters This API has standard parameters (see “Standard Parameters” on page 5) and the following parameters:

- **PIO_NAMING** (Mandatory) This is a parameter of table type CDR_NAMING_VERSIONS_OBJ_TYPE that contains CDR Naming Version attributes. Enter $OBJTYPES$TABLECNSTR for OBJECT_TYPE_RC.

- **PIO_CONSTRAINT** (Mandatory) This is a parameter of table type CDR_TABLE_CONS_OBJ_TYPE. You must enter values for CONSTRAINT_TYPE_RC.

  The possible values are: $CONSTRAINTYPES$CHECK, $CONSTRAINTYPES$NUINDEX, $CONSTRAINTYPES$PRIMARYKEY, and $CONSTRAINTYPES$UNIQUE

- **PI_CONSTRAINTCOLUMNS** (Mandatory) This is a collection of table type CDR_TABLE_CONCOLS_OBJ_TYPE. Identify the Table and the Table’s columns where you want to apply the Constraints. Depending on the Constraint, you must also provide values for attributes that define Foreign Key, or that identify the List of Values object to store the values for a CHECK Constraint.

  The following attributes are required: TABC_COMPANY_ID,TABC_OBJ_ID,TABC_OBJ_VER,FK_COL_COMPANY_ID,FK_COL_OBJ_ID,FK_COL_OBJ_VER,POSITION,COL_COMPANY_ID,COL_OBJ_ID,COL_OBJ_VER,LOV_COMPANY_ID,LOV_ID,LOV_VER

- **PI_VALS** (Optional) This is a collection of CDR_VAL_OBJ_TYPE that contains the values for a CHECK Constraint.

13.1.6 Modify a Table Definition

Use this API to modify a Table definition. You need to check out the Table definition that you want to modify.

Name CDR_PUB_DF_TABLE.ModifyTableDefinition

Signature

PROCEDURE MODIFYTABLEDEFINITION(
  P_API_VERSION IN NUMBER,
  P_INIT_MSG_LIST IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_COMMIT IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_VALIDATION_LEVEL IN NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
  X_RETURN_STATUS OUT VARCHAR2,
  X_MSG_COUNT OUT NUMBER,
  X_MSG_DATA OUT VARCHAR2,
  PIO_NAMING IN OUT CDR_NAMING_VERSIONS_OBJ_TYPE,
  PIO_TABLE IN OUT CDR_TABLE_OBJ_TYPE
);
Table definition and enter new values for the attributes you want to modify. All attributes are required.

**Note:** Use separate APIs for modifying the validation status and the version label: CDR_PUB_VL_VALIDATION.UPDATE_VAL_STATUS and CDR_PUB_DF_NAMING.UPDATE_VERSION_LABEL.

- **PIO_TABLE** (Mandatory) This is a parameter of table type CDR_TABLE_OBJ_TYPE that contains attributes specific to Table definitions. Enter the values that you want to change.

### 13.1.7 Modify a Table Descriptor

Use this API to modify a Table Descriptor. You need to check out the parent object of the Table Descriptor in order to modify it.

**Name** CDR_PUB_DF_TABLE.ModifyTableDescriptor

**Signature**

PROCEDURE MODIFYTABLEDESCRIPTOR(
  P_API_VERSION IN NUMBER,
  P_INIT_MSG_LIST IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_COMMIT IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_VALIDATION_LEVEL IN NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
  X_RETURN_STATUS OUT VARCHAR2,
  X_MSG_COUNT OUT NUMBER,
  X_MSG_DATA OUT VARCHAR2,
  PIO_NAMING IN OUT CDR_NAMING_VERSION_OBJ_TYPE,
  PIO_TABLE_DESCRIPTOR IN OUT CDR_TABLE_DESC_OBJ_TYPE
);

**Parameters**  This API has standard parameters (see "Standard Parameters" on page 5) and the following parameters:

- **PIO_NAMING** (Mandatory) This is a parameter of table type CDR_NAMING_VERSION_OBJ_TYPE that contains object attributes. Enter values to identify the Table Descriptor and enter new values for the attributes you want to modify. All attributes are required.

  **Note:** Use separate APIs for modifying the validation status and the version label: CDR_PUB_VL_VALIDATION.UPDATE_VAL_STATUS and CDR_PUB_DF_NAMING.UPDATE_VERSION_LABEL.

- **PIO_TABLE_DESCRIPTOR** (Mandatory) This is a parameter of table type CDR_TABLE_OBJ_TYPE that contains attributes specific to Table Descriptors. Enter the values that you want to change.

### 13.1.8 Modify a Table Instance

Use this API to modify a Table Instance.

**Name** CDR_PUB_DF_TABLE.ModifyTableInstance

**Signature**
PROCEDURE MODIFYTABLEINSTANCE(
  P_API_VERSION IN NUMBER,
  P_INIT_MSG_LIST IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_COMMIT IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_VALIDATION_LEVEL IN NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
  X_RETURN_STATUS OUT VARCHAR2,
  X_MSG_COUNT OUT NUMBER,
  X_MSG_DATA OUT VARCHAR2,
  PIO_NAMING IN OUT CDR_NAMING_VERSION_OBJ_TYPE,
  PIO_TABLE_INSTANCE IN OUT CDR_TABLE_REF_OBJ_TYPE
);

Parameters This API has standard parameters (see "Standard Parameters" on page 5) and the following parameters:

- **PIO_NAMING** (Mandatory) This is a parameter of table type CDR_NAMING_VERSION_OBJ_TYPE that contains object attributes. Enter values to identify the Table instance and enter new values for the attributes you want to modify. All attributes are required.

- **PIO_TABLE_INSTANCE** (Mandatory) This is a parameter of table type CDR_TABLE_OBJ_TYPE that contains attributes specific to Table instances. Enter the values that you want to change.

Note: Use separate APIs for modifying the validation status and the version label: CDR_PUB_VL_VALIDATION.UPDATE_VAL_STATUS and CDR_PUB_DF_NAMING.UPDATE_VERSIONLABEL.

13.1.9 Reorder a Column

Use this API to reorder a Table's columns. You need to check out the Table definition whose columns you want to reorder.

Name CDR_PUB_DF_TABLE.ReorderColumns

Signature
PROCEDURE REORDERCOLUMNS(
  P_API_VERSION IN NUMBER,
  P_INIT_MSG_LIST IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_COMMIT IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_VALIDATION_LEVEL IN NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
  X_RETURN_STATUS OUT VARCHAR2,
  X_MSG_COUNT OUT NUMBER,
  X_MSG_DATA OUT VARCHAR2,
  PI_REORDEROBJCOLL IN CDR_REORDER_OBJ_COLL
);

Parameters This API has standard parameters (see "Standard Parameters" on page 5) and the following parameters:

- **PI_REORDEROBJCOLL** (Mandatory) This is a collection of CDR_REORDER_OBJ_TYPE that contains the Column objects that you want to reorder. The value for ENTRY_NUMBER must be NULL. Add the Columns to the collection in the new order in which you want them.
13.1.10 **Upload a Table Descriptor or Column**

Use this API to upload Columns and/or a Table Descriptor.

**Name**  
CDR_PUB_DF_TABLE.UploadOperatorColumns

**Signature**

```
PROCEDURE UPLOADOPERATORCOLUMNS(
  P_API_VERSION  IN    NUMBER,
  P_INIT_MSG_LIST  IN    VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_COMMIT  IN    VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_VALIDATION_LEVEL  IN    NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
  X_RETURN_STATUS  OUT    VARCHAR2,
  X_MSG_COUNT  OUT    NUMBER,
  X_MSG_DATA  OUT    VARCHAR2,
  PI_COMPID  IN    NUMBER,
  PI_NSOBJID  IN    NUMBER,
  PI_NSOBJVER  IN    NUMBER,
  PI_OBJID  IN    NUMBER,
  PI_OBJVER  IN    NUMBER,
  PI_NVCOLL  IN    CDR_NAME_VALUE_PAIR_COLL
);
```

**Parameters**  
This API has standard parameters (see “Standard Parameters” on page 5) and the following parameters:

- **PI_COMPID** (Mandatory) Enter the COMPANY_ID of the Column or Table Descriptor.
- **PI_NSOBJID** (Mandatory) Enter the Namespace OBJ_ID of the Column or Table Descriptor.
- **PI_NSOBJVER** (Mandatory) Enter the Namespace OBJ_VER of the Column or Table Descriptor.
- **PI_OBJID** (Mandatory) Enter the OBJ_ID of the Column or Table Descriptor.
- **PI_OBJVER** (Mandatory) Enter the OBJ_VER of the Column or Table Descriptor.
- **PI_NVCOLL** (Mandatory) This is a collection of CDR_NAME_VALUE_PAIR_OBJ_TYPE that contains name value pairs for the variable in LSH database to store the uploaded Columns and/or Table Descriptor.

You must call CREATETEMPBLOB API before you use this API.

For SAS/C-PORT files, the variable name is the TMP_BLOB_ID which is defined after you call CREATETEMPBLOB API. For other uploads, the variable name is NULL.

13.1.11 **Check In a Table Object**

Use this API to check in a Table Object. Check in object depends on object type passed to it.

**Name**  
CDR_PUB_DF_TABLE.CheckIn

**Signature**

```
PROCEDURE CHECKIN(
  P_API_VERSION  IN    NUMBER,
  P_INIT_MSG_LIST  IN    VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_COMMIT  IN    VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
Parameters This API has standard parameters (see "Standard Parameters" on page 5) and the following parameters:

- **PIO_BASEOBJECT** (Mandatory) This is a parameter of table type CDR_BASE_OBJ_TYPE that contains object attributes. Enter values to identify the Table definition that you want to check in.

  The following attributes are required: COMPANY_ID, OBJ_ID, OBJ_VER, OBJECT_VERSION_NUMBER, NAMESPACE_OBJ_ID, NAMESPACE_OBJ_VER.

- **PI_COMMENT** (Optional) Enter the reason you are checking in the Table.

### 13.1.12 Remove a Single Object

Use this API to remove a single object of any of the following types: Table definition, Table instance, Table Descriptor, Column, or a Constraint.

**Name**  
CDR_PUB_DF_TABLE.Remove

**Signature**

```sql
PROCEDURE REMOVE(
P_API_VERSION  IN    NUMBER,
P_INIT_MSG_LIST  IN    VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
P_COMMIT  IN    VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
P_VALIDATION_LEVEL  IN    NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
X_RETURN_STATUS  OUT    VARCHAR2,
X_MSG_COUNT  OUT    NUMBER,
X_MSG_DATA  OUT    VARCHAR2,
PI_NAMING  IN OUT    CDR_NAMING_VERSION_OBJ_TYPE,
);```

**Parameters** This API has standard parameters (see "Standard Parameters" on page 5) and the following parameter:

- **PI_NAMING** (Mandatory) This is a collection of CDR_BASE_OBJ_TYPEs. For each object that you want to remove, initialize a CDR_BASE_OBJ_TYPE and then extend the collection.

  The following attributes are required: COMPANY_ID, OBJ_ID, OBJ_VER, OBJECT_VERSION_NUMBER, NAMESPACE_OBJ_ID, NAMESPACE_OBJ_VER.
This section contains the following topics:

- Section 14.1, "Define and Modify Parameters"
- Section 14.2, "Define Parameter Relations"

## 14.1 Define and Modify Parameters

This is a public interface for operations involving defined Parameter objects. For further information, see the chapter on Parameters in the *Oracle Life Sciences Data Hub Application Developer’s Guide*.

### 14.1.1 Create a Parameter

Use this API to create a parameter instance or definition or both the parameter instance and its definition.

**Name**  CDR_PUB_DF_PARAMETER.CreateParameter

**Signature**

```sql
PROCEDURE CREATEPARAMETER(
  P_API_VERSION IN NUMBER,
  P_INIT_MSG_LIST IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_COMMIT IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_VALIDATION_LEVEL IN NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
  X_RETURN_STATUS OUT VARCHAR2,
  X_MSG_COUNT OUT NUMBER,
  X_MSG_DATA OUT VARCHAR2,
  PIO_PARAMNAMING IN OUT CDR_NAMING_VERSION_OBJ_TYPE,
  PIO_CDRPARAMOBJTYPE IN OUT CDR_PARAMETER_OBJ_TYPE,
  PI_CREATE_OBJECT IN VARCHAR2,
  PI_INSTANCE_SUBTYPE_ID IN CDR_NAMINGS.OBJECT_SUBTYPE_ID%TYPE,
  PI_PARENTNAMING IN OUT CDR_BASE_OBJ_TYPE,
  PO_DEFCLASSIFICATIONCOLL IN CDR_CLASSIFICATIONS_COLL
);
```

**Parameters**  This API has standard parameters (see "Standard Parameters" on page 5) and the following parameters:

- **PIO_PARAMNAMING** (Mandatory) If you are creating a new instance of an existing Parameter Set definition (and instances of all the Parameters in the Parameter Set definition), enter values to identify the Parameter Set definition. For OBJECT_TYPE_RC enter $OBJTYPES$PARAMSETREF.
If you are creating a single Parameter, enter values for the Parameter definition you want to create or, if you are creating an instance of an existing Parameter definition, enter values to identify the definition you want to create an instance of. For OBJECT_TYPE_RC enter $OBJTYPES$PARAMETER if you are creating a definition only; $OBJTYPES$PARAMREF if you are creating an instance of an existing definition; and NULL if you are creating a new definition and an instance of it.

The following attributes are required: COMPANY_ID, OBJECT_TYPE_RC, NAMESPACE_OBJ_ID, NAMESPACE_OBJ_VER, NAMESPACE_START_OBJ_VER, NAMESPACE_END_OBJ_VER, OWNING_LOCATION_RC, CHECKED_OUT_FLAG_RC, CHECKED_OUT_ID, OBJECT_SUBTYPE_ID, DESCRIPTION, REF_COMPANY_ID, REF_OBJ_ID, REF_OBJ_VER, OBJECT_VERSION_NUMBER, VALIDATION_STATUS_RC, VERSION_LABEL.

- **PIO_CDRPARAMOBJTYPE** (Optional) This is a parameter of table type CDR_PARAMETER_OBJ_TYPE that contains Parameter-specific attributes. If you are creating an instance of an existing Parameter Set, do not enter any values here. If you are creating a single Parameter, enter values for the Parameter definition you want to create or, if you are creating an instance of an existing definition, enter values to identify the definition you want to create an instance of.

  The following attributes are required:

  For simple parameter, no LOV: COMPANY_ID, PROMPT, INPUT_OUTPUT_RC, READ_ONLY_FLAG_RC, VISIBLE_FLAG_RC, MANDATORY_FLAG_RC, DEFAULT_VALUE, POSITION, PARAM_TYPE_RC, AUTO_SHARE_FIELD_FLAG_RC.


- **PI_CREATE_OBJECT** (Mandatory) Enter DEFN to create a Parameter definition only; INST to create a Parameter instance only; BOTH to create a Parameter definition and an instance of it; or PARAMSET if you are creating a new instance of a Parameter Set.
- **PI_INSTANCE_SUBTYPE_ID** (Optional) If you are creating an instance of a single Parameter, enter the ID for the subtype you want to give the instance. If you are creating a Parameter Set instance, do not enter a value here.

- **PI_PARENTNAMING** (Optional) This is a parameter of table type CDR_BASE_OBJ_TYPE that contains object attributes. Enter values to identify the object (for example, the Program or Report Set) that contains the Parameter definition. The following attributes are required: COMPANY_ID, OBJ_ID, OBJ_VER. If you are creating an instance of a Parameter Set, do not enter a value here.

- **PO_DEFCLASSIFICATIONCOLL** (Optional) By default the new definition is classified according to the subtype you assigned it in the CDR_NAMING_VERSION_OBJ_TYPE. If you want to override the default classifications for one or more classification levels, use this parameter. This is a collection of CDR_CLA_OBJ_TYPEs, which have 5 attributes, including CLA_LEVEL_ID and CLASSIFICATION_ID.

  If you want the definition to inherit its classifications for a particular level from its parent, enter the classification level ID and, for the CLASSIFICATION_ID, enter 0 (zero). If you want to explicitly assign one or more terms for a particular level, initialize a CDR_CLA_OBJ_TYPE for each term, entering the classification level ID and, for the CLASSIFICATION_ID, the term ID. The PAR_attributes are not relevant to Parameters. Do not enter any values for them. If you are not creating a new definition, do not enter values here.

14.1.2 Check Out a Parameter

Use this API to check out a Parameter definition.

**Name**  CDR_PUB_DF_PARAMETER.CheckOutParameter

**Signature**

PROCEDURE CHECKOUTPARAMETER(
  P_API_VERSION IN NUMBER,
  P_INIT_MSG_LIST IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_COMMIT IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_VALIDATION_LEVEL IN NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
  X_RETURN_STATUS OUT VARCHAR2,
  X_MSG_COUNT OUT NUMBER,
  X_MSG_DATA OUT VARCHAR2,
  PIO_CDRNAMING IN OUT CDR_BASE_OBJ_TYPE,
  PI_COMMENT IN VARCHAR2,
  PI_ISINSTONLY IN VARCHAR2
);

**Parameters**  This API has standard parameters (see "Standard Parameters" on page 5) and the following parameters:

- **PIO_CDRNAMING** (Mandatory) This is a parameter of table type CDR_BASE_OBJ_TYPE that contains CDR Naming attributes. Enter values to identify the Parameter you want to check out. The following attributes are required: COMPANY_ID,OBJ_ID,OBJ_VER,OBJECT_VERSION_NUMBER,NAMESPACE_OBJ_ID,NAMESPACE_OBJ_VER.

- **PI_COMMENT** (Optional) Enter an explanation of why you are checking out the Parameter.
14.3 Check In a Parameter

Use this API to explicitly check in a Parameter definition.

**Name**  
CDR_PUB_DF_PARAMETER.CheckInParameter

**Signature**

```sql
PROCEDURE CHECKINPARAMETER(
    P_API_VERSION  IN    NUMBER,
    P_INIT_MSG_LIST  IN    VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_COMMIT  IN    VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_VALIDATION_LEVEL  IN    NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
    X_RETURN_STATUS  OUT    VARCHAR2,
    X_MSG_COUNT  OUT    NUMBER,
    X_MSG_DATA  OUT    VARCHAR2,
    PIO_CDRNAMING  IN OUT    CDR_BASE_OBJ_TYPE,
    PI_COMMENT  IN    VARCHAR2
);
```

**Parameters**  
This API has standard parameters (see "Standard Parameters" on page 5) and the following parameters:

- **PIO_CDRNAMING**  (Mandatory) This is a parameter of table type CDR_BASE_OBJ_TYPE that contains CDR Naming attributes. Enter values to identify the Parameter you want to check out. The following attributes are required: COMPANY_ID,OBJ_ID,OBJ_VER,OBJECT_VERSION_NUMBER,NAMESPACE_OBJ_ID,NAMESPACE_OBJ_VER.

- **PI_COMMENT**  (Optional) Enter a checkin comment.

14.4 Get Displayed Parameter Values

Use this API to get the Parameter value(s) that must be displayed in the submission Execution Setup in cases where the displayed value differs from the value used internally: for a Parameter with a classification list of values, this API returns a comma-separated list of the terms in the appropriate classification level (instead of the term_id used internally); for a Report Set Entry Title Parameter, the API returns the title (instead of the RSE obj_id); and for a Parameter with a look-up value, the API returns a display value; for example, 'Yes' instead of $YESNO$YES. You can also use this API to populate a default value for a Parameter in an Execution Setup.

**Name**  
CDR_PUB_DF_PARAMETER.GetDefaultCLAVvalue

**Signature**

```sql
FUNCTION GETDEFAULTCLAVALUE(
    P_API_VERSION  IN    NUMBER,
    P_INIT_MSG_LIST  IN    VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_COMMIT  IN    VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_VALIDATION_LEVEL  IN    NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
    PALLOWEDVALUE  IN    CDR_PARAMETERS.ALLOWED_VALUES_RC%TYPE,
    PDEFAULTVALUE  IN    VARCHAR2
) RETURN VARCHAR2;
```
Parameters This API has standard parameters (see “Standard Parameters” on page 5) and the following parameters:

- **P_COMMIT** (Optional) Accept the default value (FND_API.G_FALSE) to ensure that this individual API does not commit upon completion. Pass FND_API.G_TRUE to override the default behavior.
- **P_VALIDATION_LEVEL** (Optional) Accept the default value to perform full validation. No other values are currently supported.
- **PALLOWEDVALUE** (Optional) To get the allowed values for a Parameter with a classification LOV, enter $PARAMALLOWVALS$CLALOV. To get the title for a Report Set Entry, enter $PARAMALLOWVALS$ENTRYLOV. To get a look-up value or enter a default value, do not enter a value here.
- **PDEFAULTVALUE** (Optional) To get the allowed values for a Parameter with a classification LOV, enter the level ID of the Parameter’s LOV. To get a look-up value, enter $PARAMALLOWVALS$LOV. To get the title for a Report Set Entry, enter the RSE object ID. To set a default value for a Parameter in the Execution Setup, enter the string you want to serve as the default value. The API returns the string.

14.2 Define Parameter Relations

This packages contains the following procedures and functions:

- **Section 14.2.1, “Create a Parameter Relation Collection”**
- **Section 14.2.2, “Get Parameter Instances for Value Passing”**
- **Section 14.2.3, “Remove Parameter Relations”**

This is a public interface for operations related to passing values from one Parameter to another within a Report Set or Workflow. For further information, see the chapter on Parameters in the Oracle Life Sciences Data Hub Application Developer’s Guide.

14.2.1 Create a Parameter Relation Collection

Program instance contained in the Workflow and an input Parameter of another Program instance that is executed later in the Workflow.

**Name** CDR_PUB_DF_PARAM_RELATION.CreateParrelColl

**Signature**

PROCEDURE CREATEPARRELCOLL(
  P_API_VERSION IN NUMBER,
  P_INIT_MSG_LIST IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_COMMIT IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_VALIDATION_LEVEL IN NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
  X_RETURN_STATUS OUT VARCHAR2,
  X_MSG_COUNT OUT NUMBER,
  X_MSG_DATA OUT VARCHAR2,
  PIO_PARCOLL IN OUT CDR_PARAM_RELATION_COLL,
  PI_VALIDATERELATIONS IN VARCHAR := 'T'
);

Parameters This API has standard parameters (see “Standard Parameters” on page 5) and the following parameters:
Define Parameter Relations

- **PIO_PARCOLL** (Mandatory) This is a collection of CDR_PARAM_RELATION_OBJ_TYPEs.
  
  For each Parameter relation that you want to create, initialize a CDR_PARAM_RELATION_OBJ_TYPE and then extend the collection. For the SRC attributes enter information about the Parameter whose value will be passed to another Parameter. For the TGT attributes, enter information about the target Parameter that will receive its value from the source Parameter. For RELATION_TYPE enter either LINK or SHARE.

- **PIVALIDATERELATIONS** (Mandatory) Accept the default value of ‘T’ to validate the parameter relations in the collection. Enter ‘F’ to skip validation.

### 14.2.2 Get Parameter Instances for Value Passing

Use this API to get a list of Parameters that would be valid for either receiving a value from, or passing a value to, the Parameter you specify in the Report Set or Workflow you specify.

**Name**  CDR_PUB_DF_PARAM_RELATION.GetParameterRefs

**Signature**

```sql
PROCEDURE GETPARAMETERREFS(
  P_API_VERSION IN    NUMBER,
  P_INIT_MSG_LIST IN    VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_COMMIT IN    VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_VALIDATION_LEVEL IN    NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
  X_RETURN_STATUS  OUT    VARCHAR2,
  X_MSG_COUNT  OUT    NUMBER,
  X_MSG_DATA  OUT    VARCHAR2,
  PI_PSNSCOMPANYID IN    CDR_NAMINGS.COMPANY_ID%TYPE,
  PI_PSNSOBJID IN    CDR_NAMINGS.OBJ_ID%TYPE,
  PI_PSNSOBJVER IN    CDR_NAMING_VERSIONS.OBJ_VER%TYPE,
  PI_PAROBJID IN    CDR_NAMINGS.OBJ_ID%TYPE,
  PI_PAROBJVER IN    CDR_NAMING_VERSIONS.OBJ_VER%TYPE,
  PO_PARCOLL OUT    CDR_PARAM_RELATION_COLL,
  PI_SHARE IN    VARCHAR2
);
```

**Parameters**  This API has standard parameters (see "Standard Parameters" on page 5) and the following parameters:

- **PI_PSNSCOMPANYID** (Mandatory) Enter the company ID of the Report Set or Workflow.
- **PI_PSNSOBJID** (Mandatory) Enter the object ID of the Report Set or Workflow.
- **PI_PSNSOBJVER** (Mandatory) Enter the version number of the Report Set or Workflow.
- **PI_PAROBJID** (Mandatory) Enter the object ID of the Parameter.
- **PI_PAROBJVER** (Mandatory) Enter the version number of the Parameter.
- **PO_PARCOLL** This output parameter is a collection of CDR_PARAM_RELATION_OBJ_TYPEs containing Parameter instances in the Report Set or Workflow.
- **PI_SHARE** Enter SHAREDFROM to get a list of potential source Parameters that could pass their value to the Parameter you specified. Enter SHAREDTO to get a
list of potential target Parameters that could receive their value from the Parameter you specified.

### 14.2.3 Remove Parameter Relations

Use this API to delete one or more Parameter relations from a Report Set or Workflow.

**Name**  
CDR_PUB_DF_PARAM_RELATION.RemoveParrelColl

**Signature**

```sql
PROCEDURE REMOVEPARRELCOLL(
    P_API_VERSION IN NUMBER,
    P_INIT_MSG_LIST IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_COMMIT IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_VALIDATION_LEVEL IN NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
    X_RETURN_STATUS OUT VARCHAR2,
    X_MSG_COUNT OUT NUMBER,
    X_MSG_DATA OUT VARCHAR2,
    PIO_PARCOLL IN OUT CDR_PARAM_RELATION_COLL
) ;
```

**Parameters**  
This API has standard parameters (see “Standard Parameters” on page 5) and the following parameters:

**PIO_PARCOLL** (Mandatory) This is a collection of CDR_PARAM_RELATION_OBJ_TYPEs. For each Parameter relation that you want to delete, initialize a CDR_PARAM_RELATION_OBJ_TYPE and then extend the collection.

The following attributes are required: COMPANY_ID, SRC_PARENT_OBJ_ID, SRC_PARENT_OBJ_VER, SRC_PAR_REF_OBJ_ID, SRC_PAR_REF_OBJ_VER, TGT_PARENT_OBJ_ID, TGT_PARENT_OBJ_VER, TGT_PAR_REF_OBJ_ID, TGT_PAR_REF_OBJ_VER.
15
Variables

This is a public interface for Variable-related operations including creating, modifying, and removing Variables. It also includes functions for checking in and checking out Variables.

15.1 Create and Modify Variables

This section contains the following topics:

- Section 15.1.1, "Create a Variable"
- Section 15.1.2, "Check Out a Variable"
- Section 15.1.3, "Modify a Variable"
- Section 15.1.4, "Check In a Variable"
- Section 15.1.5, "Remove a Variable"

15.1.1 Create a Variable

Use this API to create a new Variable instance.

**Name**  CDR_PUB_DF_VARIABLE.CreateVariable

**Signature**

```sql
PROCEDURE CREATEVARIABLE(
    P_API_VERSION IN NUMBER,
    P_INIT_MSG_LIST IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_COMMIT IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_VALIDATION_LEVEL IN NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
    X_RETURN_STATUS OUT VARCHAR2,
    X_MSG_COUNT OUT NUMBER,
    X_MSG_DATA OUT VARCHAR2,
    PIO_NAMING IN OUT CDR_NAMING_VERSION_OBJ_TYPE,
    PIO_VARIABLE IN OUT CDR_VAR_OBJ_TYPE,
    PI_DEFCCLASSIFICATIONCOLL IN CDR_CLASSIFICATIONS_COLL
) ;
```

**Parameters**  This API has standard parameters (see "Standard Parameters" on page 5) and the following parameters:

- **PIO_NAMING** (Mandatory) This is a parameter of table type CDR_NAMING_VERSIONS_OBJ_TYPE that contains CDR Naming Version attributes.
  
  For OBJECT_TYPE_RC enter "$OBJTYPES$CDRVAR."
Create and Modify Variables

- **PIO_VARIABLE** (Mandatory) This is a parameter of table type CDR_VAR_OBJ_TYPE that contains attributes specific to Variables.

  The required attributes are: ORACLE_NAME, ORACLE_DATATYPE_RC, LENGTH, PRECISION, SAS_V8_NAME, SAS_LABEL, SAS_FORMAT, NULLABLE_FLAG_RC, DEFAULT_VALUE.

  Possible values for ORACLE_DATATYPE_RC are: $ORADATATYPES$DATE, $ORADATATYPES$NUMBER, and $ORADATATYPES$VARCHAR2.

  Possible values for NULLABLE_FLAG_RC are: $YESNO$NO, $YESNO$YES.

- **PI_DEFCLASSIFICATIONCOLL** By default, the variable is classified according to the subtype you assigned it in the CDR_NAMING_VERSION_OBJ_TYPE.

  If you want to override the default classifications for one or more classification levels, use this parameter. This is a collection of CDR_CLA_OBJ_TYPEs, which have 5 attributes, including CLA_LEVEL_ID and CLASSIFICATION_ID.

  If you want the definition to inherit its classifications for a particular level from its parent, enter the classification level ID and, for the CLASSIFICATION_ID, enter 0 (zero).

  If you want to explicitly assign one or more terms for a particular level, initialize a CDR_CLA_OBJ_TYPE for each term, entering the classification level ID and, for the CLASSIFICATION_ID, the term ID. The PAR_ attributes are not relevant to Variables. Do not enter any values for them.

  If you are not creating a new definition, do not enter values here.

15.1.2 Check Out a Variable

Use this API to check out a Variable definition or instance.

**Name** CDR_PUB_DF_VARIABLE.CheckOut

**Signature**

PROCEDURE CHECKOUT(
    P_API_VERSION IN NUMBER,
    P_INIT_MSG_LIST IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_COMMIT IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_VALIDATION_LEVEL IN NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
    X_RETURN_STATUS OUT VARCHAR2,
    X_MSG_COUNT OUT NUMBER,
    X_MSG_DATA OUT VARCHAR2,
    PIO_BASEOBJECT IN OUT CDR_BASE_OBJ_TYPE,
    PI_COMMENT IN VARCHAR2,
    PI_ISINSTONLY IN VARCHAR2
) ;

**Parameters** This API has standard parameters (see "Standard Parameters" on page 5) and the following parameters:

- **PIO_BASEOBJECT** (Mandatory) This is a parameter of table type CDR_BASE_OBJ_TYPE that contains object attributes. Enter values to identify the Variable definition that you want to check out.

  The following attributes are required: COMPANY_ID, OBJ_ID, OBJ_VER, OBJECT_VERSION_NUMBER, NAMESPACE_OBJ_ID, NAMESPACE_OBJ_VER.

- **PI_COMMENT** (Optional) Enter the reason you are checking out the Variable.
PI_ISINSTONLY (Mandatory) Enter $YESNO$NO.

15.1.3 Modify a Variable

Use this API to modify a Variable definition or instance.

---

**Note:** To modify a Variable definition, you must first check it out.

---

**Name** CDR_PUB_DF_VARIABLE.ModifyVariable

**Signature**

PROCEDURE MODIFYVARIABLE(
    P_API_VERSION  IN    NUMBER,
    P_INIT_MSG_LIST  IN    VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_COMMIT  IN    VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_VALIDATION_LEVEL  IN    NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
    X_RETURN_STATUS  OUT    VARCHAR2,
    X_MSG_COUNT  OUT    NUMBER,
    X_MSG_DATA  OUT    VARCHAR2,
    PI_NAMING  IN OUT    CDR_NAMING_VERSION_OBJ_TYPE,
    PI_VARIABLE  IN OUT    CDR_VAR_OBJ_TYPE
);

**Parameters** This API has standard parameters (see "Standard Parameters" on page 5) and the following parameters:

- **PI_NAMING** (Mandatory) This is a parameter of table type CDR_NAMING_VERSIONS_OBJ_TYPE that contains CDR Naming Version attributes. Initialize the attributes COMPANY_ID, OBJECT_ID, and OBJECT_VER.

- **PI_VARIABLE** (Mandatory) This is a parameter of table type CDR_VARS_OBJ_TYPE. Provide values for the attributes you want to modify.

15.1.4 Check In a Variable

Use this API to check in a Variable definition or instance.

**Name** CDR_PUB_DF_VARIABLE.CheckIn

**Signature**

PROCEDURE CHECKIN(
    P_API_VERSION  IN    NUMBER,
    P_INIT_MSG_LIST  IN    VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_COMMIT  IN    VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_VALIDATION_LEVEL  IN    NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
    X_RETURN_STATUS  OUT    VARCHAR2,
    X_MSG_COUNT  OUT    NUMBER,
    X_MSG_DATA  OUT    VARCHAR2,
    PIO_BASEOBJECT  IN OUT    CDR_BASE_OBJ_TYPE,
    PI_COMMENT  IN    VARCHAR2
);

**Parameters** This API has standard parameters (see "Standard Parameters" on page 5) and the following parameters:
Create and Modify Variables

- **PIO_BASEOBJECT** (Mandatory) This is a parameter of table type CDR_BASE_OBJ_TYPE that contains object attributes. Enter values to identify the Variable definition that you want to check in.

  The following attributes are required: COMPANY_ID, OBJ_ID, OBJ_VER, OBJECT_VERSION_NUMBER, NAMESPACE_OBJ_ID, NAMESPACE_OBJ_VER.

- **PI_COMMENT** (Optional) Enter the reason you are checking in the Variable.

### 15.1.5 Remove a Variable

Use this API to remove an existing variable object.

**Name**  CDR_PUB_DF_VARIABLE.RemoveVariable

**Signature**

```plaintext
PROCEDURE REMOVEVARIABLE(
  P_API_VERSION  IN    NUMBER,
  P_INIT_MSG_LIST  IN    VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_COMMIT  IN    VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_VALIDATION_LEVEL  IN    NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
  X_RETURN_STATUS  OUT    VARCHAR2,
  X_MSG_COUNT  OUT    NUMBER,
  X_MSG_DATA  OUT    VARCHAR2,
  PI_NAMING  IN OUT    CDR_NAMING_VERSION_OBJ_TYPE
)
```

**Parameters**  This API has standard parameters (see “Standard Parameters” on page 5) and the following parameter:

- **PI_NAMING** (Mandatory) This is a parameter of table type CDR_NAMING_VERSIONS_OBJ_TYPE that contains CDR Naming Version attributes. Initialize the attributes COMPANY_ID, OBJECT_ID, and OBJECT_VER.
This is a public interface for Work Area-related operations, including creating, modifying, removing, cloning, checking in and installing Work Areas. It also includes an API to copy instance objects into a Work Area and an API to install a single Program.

16.1 Define and Modify Work Areas

This section contains the following topics:

- Section 16.1.1, "Create a Work Area"
- Section 16.1.2, "Install a Work Area"
- Section 16.1.3, "Check In a Work Area"
- Section 16.1.4, "Modify a Work Area"
- Section 16.1.5, "Clone a Work Area"
- Section 16.1.6, "Copy Objects into a Work Area"
- Section 16.1.7, "Clone an Object"
- Section 16.1.8, "Remove a Work Area"
- Section 16.1.9, "Get the Usage Intent RC of a Work Area"
- Section 16.1.10, "Update a Work Area’s Usage Intent"
- Section 16.1.11, "Install a Program"

16.1.1 Create a Work Area

Use this API to create a new Work Area.

Name  CDR_PUB_DF_WORKAREA.CreateWorkArea

Signature

PROCEDURE CREATEWORKAREA(
  P_API_VERSION IN NUMBER,
  P_INIT_MSG_LIST IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_COMMIT IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_VALIDATION_LEVEL IN NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
  X_RETURN_STATUS OUT VARCHAR2,
  X_MSG_COUNT OUT NUMBER,
  X_MSG_DATA OUT VARCHAR2,
  PIO_SOURCECDRNAMING IN OUT CDR_NAMING_VERSION_OBJ_TYPE,
Define and Modify Work Areas

16.1.2 Install a Work Area

Use this API to install a Work Area and the instance objects in it that you specify. The API first tries to check in the Work Area and all the object instances included in the installation. If any object included in the installation is checked out by another user, the installation fails.

**Name**  CDR_PUB_DF_WORKAREA.InstallWAController

**Signature**

PROCEDURE INSTALLWACONTROLLER(
    P_API_VERSION IN NUMBER,
    P_INIT_MSG_LIST IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_COMMIT IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_VALIDATION_LEVEL IN NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
    X_RETURN_STATUS OUT VARCHAR2,
    X_MSG_COUNT OUT NUMBER,
    X_MSG_DATA OUT VARCHAR2,
    PIO_OWABASENAMING IN OUT CDR_BASE_OBJ_TYPE,
    PI_VINSTALLMODE IN CDR_INSTALLATIONS.INSTALLATION_MODE_RC%TYPE,
    PI_VFORCEREGEN IN CDR_INSTALLATIONS.FORCE_REGEN_FLAG_RC%TYPE,
    PI_DEFCLASSIFICATIONCOLL IN CDR_CLASSIFICATIONS_COLL
);
16.1.3 Check In a Work Area

Use this API to check in a Work Area and all the object instances it contains.

Name  CDR_PUB_DF_WORKAREA.CheckInWorkArea

Signature

PROCEDURE CHECKINWORKAREA(
  P_API_VERSION IN  NUMBER,
  P_INIT_MSG_LIST IN  VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_COMMIT IN  VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_VALIDATION_LEVEL IN  NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
  X_RETURN_STATUS OUT  VARCHAR2,
  X_MSG_COUNT OUT  NUMBER,
  X_MSG_DATA OUT  VARCHAR2,
);
PIO_BASENAMING IN OUT CDR_BASE_OBJ_TYPE,
PI_COMMENT IN VARCHAR2
);

Parameters  This API has standard parameters (see "Standard Parameters" on page 5) and the following parameters:

- **PIO_BASENAMING** (Mandatory) This is a parameter of table type CDR_BASE_OBJ_TYPE that contains object attributes. Enter values to identify the Work Area that you want to check in.

  The following attributes are required: COMPANY_ID, OBJECT_ID, OBJECT_VER, NAMESPACE_OBJ_ID, NAMESPACE_OBJ_VER, OBJECT_VERSION_NUMBER.

- **PI_COMMENT** (Optional) Enter the reason you are checking in the Work Area.

### 16.1.4 Modify a Work Area

Use this API to modify the name and/or description of an existing Work Area. (Use CDR_PUB_DF_WORKAREA.UPDATEUSAGEINTENT to modify a Work Area's Usage Intent attribute.)

Name  CDR_PUB_DF_WORKAREA.ModifyWorkArea

Signature

PROCEDURE MODIFYWORKAREA(
  P_API_VERSION IN NUMBER,
  P_INIT_MSG_LIST IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_COMMIT IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_VALIDATION_LEVEL IN NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
  X_RETURN_STATUS OUT VARCHAR2,
  X_MSG_COUNT OUT NUMBER,
  X_MSG_DATA OUT VARCHAR2,
  PIO_SOURCECDRNAMING IN OUT CDR_NAMING_VERSION_OBJ_TYPE,
  PIO_WORKAREAOBJTYPE IN OUT CDR_WORKAREA_OBJ_TYPE
);

Parameters  This API has standard parameters (see "Standard Parameters" on page 5) and the following parameters:

- **PIO_SOURCECDRNAMING** (Mandatory) This is a parameter of table type CDR_NAMING_VERSION_OBJ_TYPE that contains object attributes. Enter values to identify the Work Area that you want to modify and the values you want to change. All attributes are required.

- **PIO_WORKAREAOBJTYPE** (Mandatory) This is a parameter of table type CDR_WORKAREA_OBJ_TYPE that contains object attributes specific to Work Areas. Enter values to identify the Work Area that you want to modify and enter new values for the attributes you want to modify.

### 16.1.5 Clone a Work Area

Use this API to clone a Work Area. If you specify another Work Area as the target, this API overwrites that Work Area with the source Work Area. If you specify an Application Area as the target, this API creates a new Work Area identical to the source Work Area in the Application Area you specify.

Name  CDR_PUB_DF_WORKAREA.CloneWorkArea
Define and Modify Work Areas

**Signature**

```sql
PROCEDURE CLONEWORKAREA(
    P_API_VERSION  IN    NUMBER,
    P_INIT_MSG_LIST  IN    VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_COMMIT  IN    VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_VALIDATION_LEVEL  IN    NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
    X_RETURN_STATUS  OUT    VARCHAR2,
    X_MSG_COUNT  OUT    NUMBER,
    X_MSG_DATA  OUT    VARCHAR2,
    PIO_SOURCEOBJECT  IN OUT    CDR_BASE_OBJ_TYPE,
    PIO_TARGETOBJECT  IN OUT    CDR_BASE_OBJ_TYPE,
    PI_VLABEL  IN    CDR_WORKAREAS.LABEL%TYPE,
    PI_VUSAGEINTENTRC  IN    CDR_WORKAREAS.USAGE_INTENT_RC%TYPE
);
```

**Parameters**  This API has standard parameters (see “Standard Parameters” on page 5) and the following parameters:

- **PIO_SOURCEOBJECT** (Mandatory) This is a parameter of table type CDR_BASE_OBJ_TYPE that contains object attributes. Enter values to identify the Work Area that you want to clone.

  The following attributes are required: COMPANY_ID, OBJECT_ID, OBJECT_VER, NAMESPACE_OBJ_ID, NAMESPACE_OBJ_VER, OBJECT_VERSION_NUMBER.

- **PIO_TARGETOBJECT** (Mandatory) This is a parameter of table type CDR_BASE_OBJ_TYPE that contains object attributes. Enter values to identify the Work Area or Application Area to serve as the target of the clone. If you specify a Work Area, this API overwrites it with the source Work Area. If you specify an Application Area, this API creates a new Work Area identical to the source Work Area in the Application Area.

  The following attributes are required: COMPANY_ID, OBJ_ID, OBJ_VER, OBJECT_VERSION_NUMBER, NAMESPACE_OBJ_ID, NAMESPACE_OBJ_VER.

- **PI_VLABEL** (Mandatory) Enter text for the label you want to assign to the source and target Work Areas.

- **PI_VUSAGEINTENTRC** (Optional) Enter a value for the Usage Intent attribute of the target Work Area. If you do not enter a value, the API assigns the Usage Intent value of the source Work Area to the target Work Area.

### 16.1.6 Copy Objects into a Work Area

Use this API to copy one or more objects into a Work Area.

**Name**  CDR_PUB_DF_WORKAREA.CopyObjectsIntoWA

**Signature**

```sql
PROCEDURE COPYOBJECTSINTOWA(
    P_API_VERSION  IN    NUMBER,
    P_INIT_MSG_LIST  IN    VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_COMMIT  IN    VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_VALIDATION_LEVEL  IN    NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
    X_RETURN_STATUS  OUT    VARCHAR2,
    X_MSG_COUNT  OUT    NUMBER,
    X_MSG_DATA  OUT    VARCHAR2,
    PI_CDRPRREFOBJCOLL  IN    CDR_PRREF_OBJ_COLL,
    PI_CDRTARGETCONTAINEROBJECT  IN OUT    CDR_PRREF_OBJ_TYPE,
    ...
);
```
PI_CHECKINFLAG IN VARCHAR,
PI_COPYPRGASSGNMT IN VARCHAR
);

**Parameters**  This API has standard parameters (see "Standard Parameters" on page 5) and the following parameters:

- **PI_CDRPRREFOBJCOLL** (Mandatory) This is a collection of Prref objects that contains information about the instance objects you want to copy into the Work Area.

  Provide values for the attributes COMPANY_ID, OBJ_ID, OBJ_VER, PRREF_ID, PRREF_VER, NAMESPACE_OBJ_ID, NAMESPACE_OBJ_VER,OBJECT_VERSION_NUMBER for each instance object. The PRREF_ID and PRREF_VER attributes store the hierarchy information for objects.

- **PI_CDRTARGETCONTAINEROBJECT** (Mandatory) This is a parameter of Prref object type that describes the target Work Area.

  Enter values for the attributes COMPANY_ID, OBJ_ID, OBJ_VER, PRREF_ID, PRREF_VER, NAMESPACE_OBJ_ID, NAMESPACE_OBJ_VER,OBJECT_VERSION_NUMBER.

- **PI_CHECKINFLAG** (Mandatory) Enter 'Y' for this flag.

- **PI_COPYPRGASSGNMT** (Optional) This parameter is not applicable.

### 16.1.7 Clone an Object

Use this API to clone one or more instance objects in a Work Area.

**Name**  CDR_PUB_DF_WORKAREA.CloneObjects

**Signature**

PROCEDURE CLONEOBJECTS(
  P_API_VERSION  IN    NUMBER,
  P_INIT_MSG_LIST  IN    VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_COMMIT  IN    VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_VALIDATION_LEVEL  IN    NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
  X_RETURN_STATUS  OUT    VARCHAR2,
  X_MSG_COUNT  OUT    NUMBER,
  X_MSG_DATA  OUT    VARCHAR2,
  PI_SRCINSTBASEOBJCOLL  IN OUT    CDR_BASE_OBJ_COLL,
  PI_TGTWABASEOBJTYPE  IN OUT    CDR_BASE_OBJ_TYPE
);

**Parameters**  This API has standard parameters (see "Standard Parameters" on page 5) and the following parameters:

- **PI_SRCINSTBASEOBJCOLL** (Mandatory) This is a collection of base objects that you want to clone in the Work Area.

  Enter the attributes COMPANY_ID, OBJ_ID, OBJ_VER, PRREF_ID, PRREF_VER, NAMESPACE_OBJ_ID, NAMESPACE_OBJ_VER,OBJECT_VERSION_NUMBER for each instance object.

- **PI_TGTWABASEOBJTYPE** (Mandatory) This is a parameter of base object type that describes the Work Area.

  Enter the Work Area’s COMPANY_ID, OBJ_ID, OBJ_VER, NAMESPACE_OBJ_ID, NAMESPACE_OBJ_VER,OBJECT_VERSION_NUMBER.
16.1.8 Remove a Work Area

Use this API to remove a Work Area and all the instance objects it contains.

**Name**  CDR_PUB_DF_WORKAREA.RemoveWorkArea

**Signature**

PROCEDURE REMOVEWORKAREA(
    P_API_VERSION IN NUMBER,
    P_INIT_MSG_LIST IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_COMMIT IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_VALIDATION_LEVEL IN NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
    X_RETURN_STATUS OUT VARCHAR2,
    X_MSG_COUNT OUT NUMBER,
    X_MSG_DATA OUT VARCHAR2,
    PIO_SOURCECDRNAMING IN OUT CDR_NAMING_VERSION_OBJ_TYPE
);

**Parameters**  This API has standard parameters (see "Standard Parameters" on page 5) and the following parameters:

- **PIO_SOURCECDRNAMING** (Mandatory) This is a parameter of table type CDR_NAMING_VERSION_OBJ_TYPE that contains object attributes. Enter values to identify the Work Area that you want to remove. The following attributes are required: COMPANY_ID, OBJ_ID, OBJ_VER, OBJECT_VERSION_NUMBER, NAMESPACE_OBJ_ID, NAMESPACE_OBJ_VER.

16.1.9 Get the Usage Intent RC of a Work Area

Use this API to retrieve the current value of the Usage Intent attribute for a particular Work Area.

**Name**  CDR_PUB_DF_WORKAREA.GetUsageIntentRC

**Signature**

PROCEDURE GETUSAGEINTENTRC(
    P_API_VERSION IN NUMBER,
    P_INIT_MSG_LIST IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_COMMIT IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_VALIDATION_LEVEL IN NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
    X_RETURN_STATUS OUT VARCHAR2,
    X_MSG_COUNT OUT NUMBER,
    X_MSG_DATA OUT VARCHAR2,
    PI_COMPANY_ID IN CDR_NAMINGS.COMPANY_ID%TYPE,
    PI_OBJ_ID IN CDR_NAMINGS.OBJ_ID%TYPE,
    PI_OBJ_VER IN CDR_NAMING_VERSIONS.OBJ_VER%TYPE,
    PO_USAGEINTENT OUT CDR_WORKAREAS.USAGE_INTENT_RC%TYPE
);

**Parameters**  This API has standard parameters (see "Standard Parameters" on page 5) and the following parameters:

- **PI_COMPANY_ID** (Mandatory) Enter company ID
- **PI_OBJ_ID** (Mandatory) Enter the object ID of the Work Area.
- **PI_OBJ_VER** (Mandatory) Enter the object version (OBJ_VER) of the Work Area.
Define and Modify Work Areas

16.1.10 Update a Work Area’s Usage Intent

Use this API to modify the value of the Usage Intent attribute of a Work Area.

**Name**  CDR_PUB_DF_WORKAREA.UpdateUsageIntent

**Signature**

```sql
PROCEDURE UPDATEUSAGEINTENT(
    P_API_VERSION   IN   NUMBER,
    P_INIT_MSG_LIST IN   VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_COMMIT       IN   VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_VALIDATION_LEVEL IN   NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
    X_RETURN_STATUS OUT   VARCHAR2,
    X_MSG_COUNT OUT    NUMBER,
    X_MSG_DATA OUT    VARCHAR2,
    PIO_SOURCECDRNAMING IN OUT   CDR_NAMING_VERSION_OBJ_TYPE,
    PIO_WORKAREAOBJTYPE IN OUT   CDR_WORKAREA_OBJ_TYPE
);```

**Parameters**  This API has standard parameters (see "Standard Parameters" on page 5) and the following parameters:

- **PIO_SOURCECDRNAMING** (Mandatory) This is a parameter of table type CDR_NAMING_VERSION_OBJ_TYPE that contains object attributes. Enter values to identify the Work Area whose Usage Intent attribute you want to modify.
  
  The following attributes are required: COMPANY_ID,OBJ_ID,OBJ_VER.

- **PIO_WORKAREAOBJTYPE** (Mandatory) This is a parameter of table type CDR_WORKAREA_OBJ_TYPE that contains object attributes specific to Work Areas. For USAGE_INTENT_RC, enter the new value for the Usage Intent attribute.
  
  The allowed values are: $SYSVALDNSTEPS$DEVELOPMENT, $SYSVALDNSTEPS$PRODUCTION, $SYSVALDNSTEPS$QUALITYCONTROL.

16.1.11 Install a Program

Use this API to install a single Program and the Table instances to which it is mapped. You must install a Program and its source Table instances before you can work on it in an Integrated Development Environment (IDE).

**Name**  CDR_PUB_DF_WORKAREA.InstallIDEcomponents

**Signature**

```sql
PROCEDURE INSTALLIDECOMPONENTS(
    P_API_VERSION   IN   NUMBER,
    P_INIT_MSG_LIST IN   VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_COMMIT       IN   VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_VALIDATION_LEVEL IN   NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
    X_RETURN_STATUS OUT   VARCHAR2,
    X_MSG_COUNT OUT    NUMBER,
    X_MSG_DATA OUT    VARCHAR2,
    PI_NCOMPANYID IN    NUMBER,
    PI_NWAOBJID IN   NUMBER,
    PI_NWAOBJVER IN   NUMBER,
);```
Define and Modify Work Areas

Parameters  This API has standard parameters (see "Standard Parameters" on page 5) and the following parameters:

- **PI_NCOMPANYID** (Mandatory) Enter your company ID. To get your company ID, use CDR_PUB_DEF_FACTORY_UTILS.GetCompanyId.
- **PI_NWAOBJID** (Mandatory) Enter the object ID of the Work Area where the Program instance is located.
- **PI_NWAOBJVER** (Mandatory) Enter the object version (OBJ_VER) of the Work Area.
- **PI_NPRREFID** (Mandatory) Enter the object ID of the Program instance that you want to install.
- **PI_NPRREFVER** PrrefVer of the Program Instance to be installed
- **PO_VINSTLOGTYPE** This output parameter returns one of the following values to indicate whether the installation ended with a status of Warning (G_RET_IDE_INST_WARNING) or Success (G_RET_IDE_INST_INFO).
- **PO_VINSTLOG** This output parameter returns the installation log file (maximum length 32000 characters).

```
PI_NPRREFID IN NUMBER,
PI_NPRREFVER IN NUMBER,
PO_VINSTLOGTYPE OUT VARCHAR2,
PO_VINSTLOG OUT VARCHAR2
);
```
This is a public interface for Workflow-related operations, including creating, modifying, and removing Workflows.

17.1 Create and Modify Workflows

This section contains the following topics:

- Section 17.1.1, "Create a Workflow"
- Section 17.1.2, "Check Out a Workflow Definition"
- Section 17.1.3, "Create a Workflow Transition"
- Section 17.1.4, "Create a Workflow Structure Instance"
- Section 17.1.5, "Modify a Workflow"
- Section 17.1.6, "Modify a Workflow"
- Section 17.1.7, "Check In a Workflow Definition"
- Section 17.1.8, "Remove a Transition"
- Section 17.1.9, "Remove a Workflow Activity"
- Section 17.1.10, "Remove a Workflow Instance"
- Section 17.1.11, "Remove a Workflow Definition"

17.1.1 Create a Workflow

Use this API to create a Workflow definition or instance.

**Name**  CDR_PUB_DF_WORKFLOW.CreateWorkFlow

**Signature**

```
PROCEDURE CREATEWORKFLOW(
    P_API_VERSION IN NUMBER,
    P_INIT_MSG_LIST IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_COMMIT IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_VALIDATION_LEVEL IN NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
    X_RETURN_STATUS OUT VARCHAR2,
    X_MSG_COUNT OUT NUMBER,
    X_MSG_DATA OUT VARCHAR2,
    PIO_SOURCECDRNAMING IN OUT CDR_NAMING_VERSION_OBJ_TYPE,
    PI_CREATEOBJECT IN VARCHAR2,
    PI_INSTANCE_SUBTYPE_ID IN CDR_NAMINGS.OBJECT_SUBTYPE_ID%TYPE,
```
Create and Modify Workflows

PI_DEFCLASSIFICATIONCOLL IN CDR_CLASSIFICATIONS_COLL,
PI_INSTCLASSIFICATIONCOLL IN CDR_CLASSIFICATIONS_COLL
);

Parameters

This API has standard parameters (see "Standard Parameters" on page 5) and the following parameters:

- **PIO_SOURCEDRNAME** (Mandatory) This is a parameter of table type CDR_NAMING_VERSIONS_OBJ_TYPE that contains CDR Naming Version attributes.

  For OBJECT_TYPE_RC enter $OBJTYPES$WORKFLOW if you are creating a definition only; $OBJTYPES$WORKFLOWREF if you are creating an instance of an existing definition; and also if you are creating a new definition and an instance of it.

- **PI_CREATEOBJECT** (Mandatory) Enter DEFN to create a definition only; INST to create an instance of an existing definition; or BOTH to create a new definition and an instance of it.

  Valid parameters are: Definition—DEFN, instance—INST, both—BOTH.

- **PI_INSTANCE_SUBTYPE_ID** (Optional) If you are creating a new instance, enter the ID for the subtype you want to give the instance.

  If you are creating a definition only, do not enter a value for this parameter.

- **PI_DEFCLASSIFICATIONCOLL** (Optional) By default the new definition is classified according to the subtype you assigned it in the CDR_NAMING_VERSIONS_OBJ_TYPE.

  If you want to override the default classifications for one or more classification levels, use this parameter. This is a collection of CDR_CLA_OBJ_TYPEs, which have 5 attributes, including CLA_LEVEL_ID and CLASSIFICATION_ID.

  If you want the definition to inherit its classifications for a particular level from its parent, enter the classification level ID and, for the CLASSIFICATION_ID, enter 0 (zero). If you want to explicitly assign one or more terms for a particular level, initialize a CDR_CLA_OBJ_TYPE for each term, entering the classification level ID and, for the CLASSIFICATION_ID, the term ID. The PAR_attributes are not relevant to Workflow definitions. Do not enter any values for them.

- **PI_INSTCLASSIFICATIONCOLL** (Optional) By default the new instance is classified according to the subtype you assigned it in the PI_INSTANCE_SUBTYPE_ID.

  If you want to override the default classifications for one or more classification levels, use this parameter. This is a collection of CDR_CLA_OBJ_TYPEs, which have 5 attributes, including CLA_LEVEL_ID and CLASSIFICATION_ID.

  If you want the instance to inherit its classifications for a particular level from its parent Work Area, enter the classification level ID and, for the CLASSIFICATION_ID, enter 0 (zero).

  If you want to explicitly assign one or more terms for a particular level, initialize a CDR_CLA_OBJ_TYPE for each term, entering the classification level ID and, for the CLASSIFICATION_ID, the term ID. The PAR_attributes are not relevant to Workflows. Do not enter any values for them. If you are not creating a new instance, do not enter values here.
17.1.2 Check Out a Workflow Definition

Use this API to check out a Workflow definition.

**Name**  
CDR_PUB_DF_WORKFLOW.CheckOutWorkFlow

**Signature**

PROCEDURE CHECKOUTWORKFLOW(
  P_API_VERSION IN NUMBER,
  P_INIT_MSG_LIST IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_COMMIT IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_VALIDATION_LEVEL IN NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
  X_RETURN_STATUS OUT VARCHAR2,
  X_MSG_COUNT OUT NUMBER,
  X_MSG_DATA OUT VARCHAR2,
  PIO_CDRWORKFLOW IN OUT CDR_BASE_OBJ_TYPE,
  PI_COMMENT IN VARCHAR2,
  PI_ISINSTONLY IN VARCHAR2,
  PI_OPTYPE IN VARCHAR2
);

**Parameters**  
This API has standard parameters (see “Standard Parameters” on page 5) and the following parameters:

- **PIO_CDRWORKFLOW** [Mandatory] This is a parameter of table type CDR_BASE_OBJ_TYPE that contains object attributes. Enter values to identify the Workflow definition that you want to check in.
  The following attributes are mandatory: COMPANY_ID, OBJECT_ID, OBJECT_VER, NAMESPACE_OBJ_ID, NAMESPACE_OBJ_VER, OBJECT_VERSION_NUMBER. Initialize these attributes in CDR_BASE_OBJ_TYPE.

- **PI_COMMENT** [Optional] Enter the reason for checking out the Workflow definition.

- **PI_ISINSTONLY** (Mandatory) Enter $YESNO$NO. (The $YESNO$YES setting is used internally only.)

- **PI_OPTYPE** (Mandatory) Enter NULL for this parameter.

17.1.3 Create a Workflow Transition

Use this API to create a Workflow transition. A Workflow transition connects two Workflow activities.

**Name**  
CDR_PUB_DF_WORKFLOW.CreateWfTransition

**Signature**

PROCEDURE CREATEWFTRANSITION(
  P_API_VERSION IN NUMBER,
  P_INIT_MSG_LIST IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_COMMIT IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_VALIDATION_LEVEL IN NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
  X_RETURN_STATUS OUT VARCHAR2,
  X_MSG_COUNT OUT NUMBER,
  X_MSG_DATA OUT VARCHAR2,
  PIO_SOURCECDRNAMING IN OUT CDR_NAMING_VERSION_OBJ_TYPE,
  PI_CDRWFTRANSITION IN OUT CDR_WMG_TRANS_OBJ_TYPE
);
Parameters  This API has standard parameters (see "Standard Parameters" on page 5) and the following parameters:

- **PIO_SOURCECDRNAMING** (Mandatory) This is a parameter of table type CDR_NAMING_VERSIONS_OBJ_TYPE that contains CDR Naming Version attributes.

For OBJECT_TYPE_RC enter $OBJTYPES$WORKFLOW if you are creating a definition only; $OBJTYPES$WORKFLOWREF if you are creating an instance of an existing definition; and also if you are creating a new definition and an instance of it.

- **PI_CDRWFTRANSITION** The Object ID and Version of the two Workflow activities being connected through this transition should be populated in the CDR_WMG_TRANS_OBJ_TYPE.

In the CONDITION_RC attribute, specify the activity condition based on which the workflow transitions to the other specified activity.

The possible values are $WFTRANSITIONS$NONE, $WFTRANSITIONS$SUCCESS, $WFTRANSITIONS$ERROR,$WFTRANSITIONS$WARNING

### 17.1.4 Create a Workflow Structure Instance

Use this API to create a Workflow Structure. These are the Workflow structures: And, Or, Start, End_Success, End_Warning, End_Error, and Fork.

**Name**  CDR_PUB_DF_WORKFLOW.CreateWfStructref

**Signature**

```sql
PROCEDURE CREATEWFSTRUCTREF(
P_API_VERSION  IN    NUMBER,
P_INIT_MSG_LIST  IN    VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
P_COMMIT  IN    VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
P_VALIDATION_LEVEL  IN    NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
X_RETURN_STATUS  OUT    VARCHAR2,
X_MSG_COUNT  OUT    NUMBER,
X_MSG_DATA  OUT    VARCHAR2,
PIO_SOURCECDRNAMING  IN OUT    CDR_NAMING_VERSION_OBJ_TYPE
);
```

**Parameters**  This API has standard parameters (see "Standard Parameters" on page 5) and the following parameter:

- **PIO_SOURCECDRNAMING** (Mandatory) This is a parameter of table type CDR_NAMING_VERSIONS_OBJ_TYPE that contains CDR Naming Version attributes.

For OBJECT_TYPE_RC enter $OBJTYPES$WORKFLOW if you are creating a definition only; $OBJTYPES$WORKFLOWREF if you are creating an instance of an existing definition; and also if you are creating a new definition and an instance of it.

### 17.1.5 Modify a Workflow

**Name**  CDR_PUB_DF_WORKFLOW.ModifyWorkFlow

**Signature**

```sql
PROCEDURE MODIFYWORKFLOW(
);
```
Create and Modify Workflows

Parameters

This API has standard parameters (see "Standard Parameters" on page 5) and the following parameters:

PIO_SOURCECDRNAMING (Mandatory) This is a parameter of table type CDR_NAMING_VERSIONS_OBJ_TYPE that contains CDR Naming Version attributes.

For OBJECT_TYPE_RC enter $OBJTYPES$WORKFLOW if you are creating a definition only; $OBJTYPES$WORKFLOWREF if you are creating an instance of an existing definition; and also if you are creating a new definition and an instance of it.

17.1.6 Modify a Workflow

Use this API to reorder workflow transitions.

Name
CDR_PUB_DF_WORKFLOW.ReorderWfTransitions

Signature

PROCEDURE REORDERWFTRANSITIONS(
  P_API_VERSION IN NUMBER,
  P_INIT_MSG_LIST IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_COMMIT IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_VALIDATION_LEVEL IN NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
  X_RETURN_STATUS OUT VARCHAR2,
  X_MSG_COUNT OUT NUMBER,
  X_MSG_DATA OUT VARCHAR2,
  PI_REORDEROBJCOLL IN CDR_REORDER_OBJ_COLL
);

Parameters

This API has standard parameters (see "Standard Parameters" on page 5) and the following parameter:

PI_REORDEROBJCOLL Collection of the columns to be re-ordered.

17.1.7 Check In a Workflow Definition

Use this API to check in a Workflow definition object.

Name
CDR_PUB_DF_WORKFLOW.CheckInWorkFlowDef

Signature

PROCEDURE CHECKINWORKFLOODEF(
  P_API_VERSION IN NUMBER,
  P_INIT_MSG_LIST IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_COMMIT IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_VALIDATION_LEVEL IN NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
  X_RETURN_STATUS OUT VARCHAR2,
  X_MSG_COUNT OUT NUMBER,
  X_MSG_DATA OUT VARCHAR2,
PIO_CDRWORKFLOW IN OUT CDR_BASE_OBJ_TYPE,
PI_COMMENT IN VARCHAR2
);

**Parameters** This API has standard parameters (see "Standard Parameters" on page 5) and the following parameters:

- **PIO_CDRWORKFLOW** This parameter refers to the workflow definition to be checked in.
  
  Initialize in cdr_base_obj_type, the basic naming details (COMPANY_ID,OBJ_ID,OBJ_VER,NAMESPACE_OBJ_ID,NAMESPACE_OBJ_VER, OBJECT_VERSION_NUMBER) of the workflow to be checked in.

- **PI_COMMENT** Comment to be associated with the check in operation

### 17.1.8 Remove a Transition

Use this API to remove one or more Workflow transitions.

**Name** CDR_PUB_DF_WORKFLOW.RemoveWfTransition

**Signature**

PROCEDURE REMOVEWFTRANSITION(
  P_API_VERSION IN NUMBER,
  P_INIT_MSG_LIST IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_COMMIT IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_VALIDATION_LEVEL IN NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
  X_RETURN_STATUS OUT VARCHAR2,
  X_MSG_COUNT OUT NUMBER,
  X_MSG_DATA OUT VARCHAR2,
  PI_CDRCHILDOBJS IN CDR_BASE_OBJ_COLL
);

**Parameters** This API has standard parameters (see "Standard Parameters" on page 5) and the following parameter:

**PI_CDRCHILDOBJS** (Mandatory) This is a collection of CDR_BASE_OBJ_TYPES.

For each Workflow transition that you want to remove, initialize a CDR_BASE_OBJ_TYPE and then extend the collection.

The following attributes are required for each Workflow transition: COMPANY_ID, OBJ_ID, OBJ_VER, NAMESPACE_OBJ_ID, NAMESPACE_OBJ_VER, OBJECT_VERSION_NUMBER.

### 17.1.9 Remove a Workflow Activity

Use this API to remove one or more Workflow activities. Workflow activities include all executable objects: Load Sets, Programs, Data Marts, and Report Sets; and also Workflow structures: And, Or, Start, End_Success, End_Warning, End_Error, and Fork.

**Name** CDR_PUB_DF_WORKFLOW.RemoveObjectsFromWorkflow

**Signature**

PROCEDURE REMOVEOBJECTSFROMWORKFLOW(
  P_API_VERSION IN NUMBER,
  P_INIT_MSG_LIST IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_COMMIT IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
Parameters This API has standard parameters (see "Standard Parameters" on page 5) and the following parameter:

PI_CDRCHILDDBOJS (Mandatory) This is a collection of CDR_BASE_OBJ_TYPEs. For each Workflow activity that you want to remove, initialize a CDR_BASE_OBJ_TYPE and then extend the collection.

The following attributes are required for each Workflow activity: COMPANY_ID, OBJ_ID, OBJ_VER, NAMESPACE_OBJ_ID, NAMESPACE_OBJ_VER, OBJECT_VERSION_NUMBER.

17.1.10 Remove a Workflow Instance

Use this API to remove a Workflow instance object.

Name CDR_PUB_DF_WORKFLOW.Remove

Signature

PROCEDURE REMOVE(
  P_API_VERSION IN NUMBER,
  P_INIT_MSG_LIST IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_COMMIT IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_VALIDATION_LEVEL IN NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
  X_RETURN_STATUS OUT VARCHAR2,
  X_MSG_COUNT OUT NUMBER,
  X_MSG_DATA OUT VARCHAR2,
  PI_CDRCHILDDBOJS IN CDR_BASE_OBJ_COLL
);

Parameters This API has standard parameters (see "Standard Parameters" on page 5) and the following parameter:

PI_BASEOBJCOLL This refers to collection of workflow instances to be removed.

17.1.11 Remove a Workflow Definition

Use this API to remove one or more Workflow definitions.

Name CDR_PUB_DF_WORKFLOW.RemoveWorkflow

Signature

PROCEDURE REMOVWORKFLOW(
  P_API_VERSION IN NUMBER,
  P_INIT_MSG_LIST IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_COMMIT IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_VALIDATION_LEVEL IN NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
  X_RETURN_STATUS OUT VARCHAR2,
  X_MSG_COUNT OUT NUMBER,
  X_MSG_DATA OUT VARCHAR2,
  PIO_CDRWORKFLOW IN OUT CDR_BASE_OBJ_TYPE
);
Parameters This API has standard parameters (see "Standard Parameters" on page 5) and the following parameter:

PIO_CDRWORKFLOW (Mandatory) This is a parameter of CDR_BASE_OBJ_TYPEs.

For each Workflow that you want to remove, initialize a CDR_BASE_OBJ_TYPE.

The following attributes are required: COMPANY_ID, OBJ_ID, OBJ_VER, NAMESPACE_OBJ_ID, NAMESPACE_OBJ_VER, OBJECT_VERSION_NUMBER of the workflow to be removed should be initialized in cdr_base_obj_type.
This is a public interface for Notifications-related operations.

18.1 Create and Modify Notifications

This section contains the following topics:

- Section 18.1.1, "Create a Notification"
- Section 18.1.2, "Create a Notification Recipient"
- Section 18.1.3, "Create a Notification Link"
- Section 18.1.4, "Check Out a Notification Definition"
- Section 18.1.5, "Modify a Notification Definition"
- Section 18.1.6, "Modify a Notification Instance"
- Section 18.1.7, "Send a Notification"
- Section 18.1.8, "Check In a Notification Definition"
- Section 18.1.9, "Remove a Notification Link"
- Section 18.1.10, "Remove a Notification Recipient"
- Section 18.1.11, "Remove a Notification"

18.1.1 Create a Notification

Use this API to create a Notification definition or instance. This API also initializes the classification of the new Notification object.

**Name** CDR_PUB_DF_NOTIFICATIONS.CreateNotification

**Signature**

```sql
PROCEDURE CREATE_NOTIFICATION(  
  P_API_VERSION  IN    NUMBER,  
  P_INIT_MSG_LIST  IN    VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,  
  P_COMMIT  IN    VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,  
  P_VALIDATION_LEVEL  IN    NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,  
  X_RETURN_STATUS  OUT    VARCHAR2,  
  X_MSG_COUNT  OUT    NUMBER,  
  X_MSG_DATA  OUT    VARCHAR2,  
  PIO_SOURCECDRNAMING  IN OUT    CDR_NAMING_VERSION_OBJ_TYPE,  
  PI_CDRNOTIFICATION  IN    CDR_NOTIFICATION_OBJ_TYPE,  
  PI_CREATEOBJECT  IN    VARCHAR2,
)"
```
18-2

Oracle Life Sciences Data Hub Application Programming Interface Guide

Create and Modify Notifications

Parameters

This API has standard parameters (see "Standard Parameters" on page 5) and the following parameters:

- **PIO_SOURCECDRNAMING** (Mandatory) This is a parameter of table type CDR_NAMING_VERSIONS_OBJ_TYPE that contains CDR Naming Version attributes.
  
  If you are creating a new definition only or a new definition and an instance of it, enter values for the new definition.
  
  If you are creating an instance of an existing definition, enter values to identify the existing definition.
  
  For OBJECT_TYPE_RC enter $OBJTYPES$NOTIFICATION if you are creating a definition only; $OBJTYPES$NOTIFREF if you are creating an instance of an existing definition; and also if you are creating a new definition and an instance of it.

- **PI_CDRNOTIFICATION** (Mandatory) This is a parameter of table type CDR_NOTIFICATION_OBJ_TYPE that contains Notification specific attributes.
  
  Enter FYI or APPROVAL for NOTIF_TYPE_RC.
  
  Enter HIGH, MEDIUM, or LOW for NOTIF_PRIORITY_RC.
  
  Enter ALL or ANY for ALL_REPLIES_FLAG_RC for Notifications of type APPROVAL.

- **PI_CREATEOBJECT** Enter DEFN for creating a definition, INST for creating an instance, and BOTH for creating a definition and an instance of it.

- **PI_DEFCLASSIFICATIONCOLL** (Optional) By default the new definition is classified according to the subtype you assigned it in the CDR_NAMING_VERSION_OBJ_TYPE.
  
  If you want to override the default classifications for one or more classification levels, use this parameter. This is a collection of CDR_CLA_OBJ_TYPEs, which have 5 attributes, including CLA_LEVEL_ID and CLASSIFICATION_ID.
  
  If you want the definition to inherit its classifications for a particular level from its parent, enter the classification level ID and, for the CLASSIFICATION_ID, enter 0 (zero).
  
  If you want to explicitly assign one or more terms for a particular level, initialize a CDR_CLA_OBJ_TYPE for each term, entering the classification level ID and, for the CLASSIFICATION_ID, the term ID.
  
  The PAR_ attributes are not relevant to Notifications. Do not enter any values for them.

18.1.2 Create a Notification Recipient

Use this API to create Notification recipients. Notification recipients are group roles of a user group.

**Name**  CDR_PUB_DF_NOTIFICATIONS.CreateNotificationRecipient

**Signature**

PROCEDURE CREATENOTIFICATIONRECIPIENT(
Create and Modify Notifications

18-3

Parameters

This API has standard parameters (see “Standard Parameters” on page 5) and the following parameter:

PI_CDRNOTIFRECIPIENTS This is a collection of CDR_NOTIF_RECIPIENT_OBJ_TYPE.

If the Notification is of type APPROVAL, for the attribute FALLBACK_FLAG_RC you need to enter $NOTIFRCPTTYPE$PRIMARY if the recipient is Primary or $NOTIFRCPTTYPE$BACKUP if the recipient is Backup.

If the Primary recipients do not respond to an APPROVAL type of Notification within the defined time frame, then the Notification is sent to the Backup recipients.

18.1.3 Create a Notification Link

Use this API to create links to Planned Outputs of executables owned by a Workflow.

Name CDR_PUB_DF_NOTIFICATIONS.CreateNotificationLinks

Signature

PROCEDURE CREATE_NOTIFICATIONLINKS(
    P_API_VERSION IN NUMBER,
    P_INIT_MSG_LIST IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_COMMIT IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_VALIDATION_LEVEL IN NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
    X_RETURN_STATUS OUT VARCHAR2,
    X_MSG_COUNT OUT NUMBER,
    X_MSG_DATA OUT VARCHAR2,
    PI_CDRNOTIFLINKS IN CDR_NTFRCP_COLL
);

Parameters

This API has standard parameters (see “Standard Parameters” on page 5) and the following parameters:

PI_CDRNOTIFLINKS This is a collection of CDR_NOTIF_LINKS_OBJ_TYPE.

You need to enter values for the following attributes: COMPANY_ID, NFINST_OBJ_ID, NFINST_OBJ_VER, PO_COMPANY_ID, PO_OBJ_ID, ACTIVITY_COMPANY_ID, ACTIVITY_OBJ_ID, ACTIVITY_OBJ_VER.

18.1.4 Check Out a Notification Definition

Use this API to check out a Notification definition.

Name CDR_PUB_DF_NOTIFICATIONS.CheckOutNotif

Signature

PROCEDURE CHECKOUTNOTIF(
    P_API_VERSION IN NUMBER,
Create and Modify Notifications

18.1.5 Modify a Notification Definition

Use this API to modify a Notification definition. You need to check out the Notification definition first.

Name  CDR_PUB_DF_NOTIFICATIONS.ModifyNotification

Signature

PROCEDURE MODIFYNOTIFICATION(
  P_API_VERSION  IN    NUMBER,
  P_INIT_MSG_LIST  IN    VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_COMMIT  IN    VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_VALIDATION_LEVEL  IN    NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
  X_RETURN_STATUS  OUT    VARCHAR2,
  X_MSG_COUNT  OUT    NUMBER,
  X_MSG_DATA  OUT    VARCHAR2,
  PIO_SOURCECDRNAMING  IN OUT    CDR_NAMING_VERSION_OBJ_TYPE,
  PI_CDRNOTIFICATION  IN    CDR_NOTIFICATION_OBJ_TYPE
);
18.1.6 Modify a Notification Instance

Use this API to modify a Notification instance.

**Name**  CDR_PUB_DF_NOTIFICATIONS.ModifyNotificationInstance

**Signature**

PROCEDURE MODIFYNOTIFICATIONINSTANCE(
  P_API_VERSION IN NUMBER,
  P_INIT_MSG_LIST IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_COMMIT IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_VALIDATION_LEVEL IN NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
  X_RETURN_STATUS OUT VARCHAR2,
  X_MSG_COUNT OUT NUMBER,
  X_MSG_DATA OUT VARCHAR2,
  PIO_SOURCECDRNAMING IN OUT CDR_NAMING_VERSION_OBJ_TYPE,
  PI_CDRNOTIFICATION IN CDR_NOTIFICATION_OBJ_TYPE);

**Parameters**  This API has standard parameters (see “Standard Parameters” on page 5) and the following parameters:

- **PIO_SOURCECDRNAMING** (Mandatory) This is a parameter of table type CDR_NAMING_VERSIONS_OBJ_TYPE that contains CDR Naming Version attributes.
  
  Initialize the attributes COMPANY_ID, OBJECT_ID, and OBJECT_VER.

- **PI_CDRNOTIFICATION** (Mandatory) This is a parameter of table type CDR_NOTIFICATION_OBJ_TYPE. Provide values for the attributes you want to modify.

18.1.7 Send a Notification

Use this API to send an information Notification either from a Program or from another database user.

**Name**  CDR_PUB_DF_NOTIFICATIONS.Send_FYI_Notification

**Signature**

PROCEDURE SEND_FYI_NOTIFICATION
  PI_JOB_ID                      NUMBER                  IN
  PI_USER_GROUP                  VARCHAR2                IN
  PI_ROLE_CODE                   VARCHAR2                IN
  PI_SUBJECT                     VARCHAR2                IN
  PI_BODY                        VARCHAR2                IN
  PI_URL                         VARCHAR2                IN
  PI_URLNAME                     VARCHAR2                IN     DEFAULT
  PI_CDRNOTIFLINKS               CDR_NOTIF_LINKS_OBJ_COLL IN     DEFAULT,
);

If the user wants to include just one URL in the notification, it can be passed in pi_url and the string to be shown for the hyperlink should be passed in pi_urlName.

On the other hand if the user wants to include links to some or all of the outputs generated by the job, then the list of the corresponding planned output IDs should be passed in the last collection parameter.

**Parameters**  This API has standard parameters (see “Standard Parameters” on page 5) and the following parameters:
Create and Modify Notifications

- **PI_JOB_ID** Enter the job ID created for the corresponding job for the notifications sent.
- **PI_USER_GROUP** Enter the user group identifier.
- **PI_ROLE_CODE** Enter the role code identifier.
- **PI_SUBJECT** Enter the subject of the notification.
- **PI_BODY** Enter the body content of the notification.
- **PI_URL** Enter the URL to be embedded in the notification.

### 18.1.8 Check In a Notification Definition

Use this API to check in a Notification definition.

**Name**  CDR_PUB_DF_NOTIFICATIONS.CheckInNotifDef

**Signature**

```sql
PROCEDURE CHECKINNOTIFDEF(
    P_API_VERSION IN NUMBER,
    P_INIT_MSG_LIST IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_COMMIT IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_VALIDATION_LEVEL IN NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
    X_RETURN_STATUS OUT VARCHAR2,
    X_MSG_COUNT OUT NUMBER,
    X_MSG_DATA OUT VARCHAR2,
    PIO_CDRNOTIF IN OUT CDR_BASE_OBJ_TYPE,
    PI_COMMENT IN VARCHAR2
);
```

**Parameters**  This API has standard parameters (see "Standard Parameters" on page 5) and the following parameters:

- **PIO_CDRNOTIF** [Mandatory] This is a parameter of table type CDR_BASE_OBJ_TYPE that object attributes.
  
  Identify the Notification definition you want to check in. Provide the basic naming attributes: COMPANY_ID, OBJECT_ID, OBJECT_VER, NAMESPACE_OBJ_ID, NAMESPACE_OBJ_VER, OBJECT_VERSION_NUMBER. Initialize these attributes in CDR_BASE_OBJ_TYPE.

- **PI_COMMENT** (Optional) Enter the reason you are checking in the Notification.

### 18.1.9 Remove a Notification Link

Use this API to remove links to Planned Outputs of executables owned by the Workflow.

**Name**  CDR_PUB_DF_NOTIFICATIONS.RemoveNotificationLinks

**Signature**

```sql
PROCEDURE REMOVENOTIFICATIONLINKS(
    P_API_VERSION IN NUMBER,
    P_INIT_MSG_LIST IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_COMMIT IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_VALIDATION_LEVEL IN NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
    X_RETURN_STATUS OUT VARCHAR2,
    X_MSG_COUNT OUT NUMBER,
```
Create and Modify Notifications

Workflow Notifications

18.10 Remove a Notification Recipient

Use this API to remove a Notification recipient.

Name  CDR_PUB_DF_NOTIFICATIONS.RemoveNotificationRecipients

Signature

PROCEDURE REMOVENOTIFICATIONRECIPIENTS(
  P_API_VERSION  IN    NUMBER,
  P_INIT_MSG_LIST  IN    VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_COMMIT  IN    VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_VALIDATION_LEVEL  IN    NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
  X_RETURN_STATUS  OUT    VARCHAR2,
  X_MSG_COUNT  OUT    NUMBER,
  X_MSG_DATA  OUT    VARCHAR2,
  PI_CDRNOTIFRECIPIENTS  IN    CDR_NTFRCP_COLL
);

Parameters  This API has standard parameters (see “Standard Parameters” on page 5) and the following parameter:

PI_CDRNOTIFRECIPIENTS  (Mandatory) This is a collection of CDR_NOTIF_RECIPIENT_OBJ_TYPE. For each Recipient that you want to remove, initialize a CDR_NOTIF_RECIPIENT_OBJ_TYPE and then extend the collection.

The following attributes are required: COMPANY_ID,OBJ_ID,OBJ_VER,USER_GROUP_ID,ROLE_ID,FALLBACK_FLAG_RC.

18.11 Remove a Notification

Use this API to remove a Notification definition.

Name  CDR_PUB_DF_NOTIFICATIONS.RemoveNotifications

Signature

PROCEDURE REMOVENOTIFICATIONS(
  P_API_VERSION  IN    NUMBER,
  P_INIT_MSG_LIST  IN    VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_COMMIT  IN    VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_VALIDATION_LEVEL  IN    NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
  X_RETURN_STATUS  OUT    VARCHAR2,
  X_MSG_COUNT  OUT    NUMBER,
  X_MSG_DATA  OUT    VARCHAR2,
  PI_BASEOBJCOLL  IN OUT    CDR_BASE_OBJ_COLL
);
Create and Modify Notifications

Parameters  This API has standard parameters (see "Standard Parameters" on page 5) and the following parameter:

**PI_BASEOBJCOLL** (Mandatory) This is a collection of CDR_BASE_OBJ_TYPE. For each Notification that you want to remove, initialize a CDR_BASE_OBJ_TYPE and then extend the collection.

The following attributes are required: COMPANY_ID,OBJ_ID,OBJ_VER,OBJECT_VERSION_NUMBER,NAMESPACE_OBJ_ID,NAMESPACE_OBJ_VER.
This part of the Oracle Life Sciences Data Hub (Oracle LSH) API guide contains APIs that you can use to perform common tasks such as creating and modifying execution setups, mapping, managing outputs, labels and jobs. This part also contains APIs that you use for classification, validation and handling security in Oracle LSH.

Part III contains the following chapters:

- Chapter 20, "Execution Setups"
- Chapter 21, "Mappings"
- Chapter 22, "Outputs"
- Chapter 23, "Version Labels"
- Chapter 24, "Classification"
- Chapter 25, "Job Execution"
- Chapter 26, "Security Policy"
- Chapter 27, "Validation"
This is a public interface for operations related to the setting up of utilities in Oracle LSH.

This section contains the following topics:

- Section 19.1, "Initialize APIs"
- Section 19.2, "Define and Modify Adapters"
- Section 19.3, "Host Definition Constants"
- Section 19.4, "Get Factory Support"
- Section 19.5, "Get Factory Utilities"
- Section 19.6, "Get Factory Validations"
- Section 19.7, "Get Data from Naming Tables"
- Section 19.8, "Read Messages"

19.1 Initialize APIs

This is a public interface that is used internally to initialize all other Oracle LSH external API packages. See "Calling the Security API Package" on page 1-3 for details. See "Code Example Using Security and Error Message APIs" on page 1-5 for an example of a program that calls the initialization API.

This section contains the following topics:

- Section 19.1.1, "Initialize a Package"
- Section 19.1.2, "Verify Whether an API is Enabled"
- Section 19.1.3, "Enable an API"
- Section 19.1.4, "Disable an API"

19.1.1 Initialize a Package

This is used internally to initialize external Oracle LSH API packages.

Name  CDR_PUB_API_INITIALIZATION.Initialization

Signature

PROCEDURE INITIALIZATION(
    P_API_VERSION  IN    NUMBER,
    P_INIT_MSG_LIST  IN    VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
19.1.2 Verify Whether an API is Enabled

Use this API to find out whether or not the API you want to use is enabled.

**Name**  
CDR_PUB_API_INITIALIZATION.IsAPInabled

**Signature**

FUNCTION ISAPINABLED(
    P_API_VERSION IN NUMBER,
    P_INIT_MSG_LIST IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_COMMIT IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_VALIDATION_LEVEL IN NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL
) RETURN BOOLEAN;

**Return**  
BOOLEAN

True, if the API is enabled or False, if it is disabled.

**Parameters**  
This API has standard parameters. See "Standard Parameters" on page 5 for details.

19.1.3 Enable an API

Use this API to enable LSH APIs that you want to use in a session. This API must be called at the beginning of each Program that uses LSH APIs.

**Name**  
CDR_PUB_API_INITIALIZATION.EnableAPIS

**Signature**

PROCEDURE ENABLEAPIS(
    P_API_VERSION IN NUMBER,
    P_INIT_MSG_LIST IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_COMMIT IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_VALIDATION_LEVEL IN NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
    X_RETURN_STATUS OUT VARCHAR2,
    X_MSG_COUNT OUT NUMBER,
    X_MSG_DATA OUT VARCHAR2
);

**Parameters**  
This API has standard parameters. See "Standard Parameters" on page 5 for details.
19.1.4 Disable an API

Use this API to disable LSH APIs that you have used in a session. Call this API at the end of each Program that uses LSH APIs.

**Name**  CDR_PUB_API_INITIALIZATION.DisableAPIs

**Signature**

```sql
PROCEDURE DisableAPIs(
  P_API_VERSION NUMBER,
  P_INIT_MSG_LIST VARCHAR2 DEFAULT,  
  P_COMMIT VARCHAR2 DEFAULT,  
  P_VALIDATION_LEVEL NUMBER DEFAULT,  
  X_RETURN_STATUS VARCHAR2 OUT,  
  X_MSG_COUNT NUMBER OUT,
  X_MSG_DATA VARCHAR2 OUT
);
```

**Return**  BOOLEAN
True, if the API is enabled or False, if it is disabled.

**Parameters**  This API has standard parameters. See "Standard Parameters" on page 5 for details.

19.2 Define and Modify Adapters

This package is used to create adapter domains and adapter areas for user defined adapters.

This section contains the following topics:
- Section 19.2.1, "Create an Adapter Domain"
- Section 19.2.2, "Modify an Adapter Domain"
- Section 19.2.3, "Create an Adapter Area"
- Section 19.2.4, "Modify an Adapter Area"
- Section 19.2.5, "Populate a Tech Type Table"
- Section 19.2.6, "Modify a Tech Type Table"

19.2.1 Create an Adapter Domain

Use this API to create an Adapter Domain.

**Name**  CDR_PUB_ATK_ADAPTER.CreateAdapterDomain

**Signature**

```sql
PROCEDURE CreateAdapterDomain(
  P_API_VERSION NUMBER,
  P_INIT_MSG_LIST VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,  
  P_COMMIT VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,  
  P_VALIDATION_LEVEL NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,  
  X_RETURN_STATUS VARCHAR2 OUT,  
  X_MSG_COUNT NUMBER OUT,
  X_MSG_DATA VARCHAR2 OUT,
  PIO_ADAPTERDOMAINNAMING IN OUT CDR_NAMING_VERSION_OBJ_TYPE
);
```


Parameters This API has standard parameters (see "Standard Parameters" on page 5) and the following parameter:

PIO_ADAPTERDOMAINNAMING (Mandatory) This is a parameter of table type CDR_NAMING_VERSION_OBJ_TYPE that contains object attributes. Enter values for the Adapter Domain that you are creating. For OBJECT_TYPE_RC enter $OBJTYPES$ADAPTERDOMAIN.

The following attributes are required: COMPANY_ID, NAME

19.2.2 Modify an Adapter Domain

Use this API to modify an Adapter Domain.

Name CDR_PUB_ATK_ADAPTER.ModifyAdapterDomain

Signature

PROCEDURE MODIFYADAPTERDOMAIN(
P_API_VERSION IN NUMBER,
P_INIT_MSG_LIST IN VARCHAR2 DEFAULT CDR_PUB_DEF_CONSTANTS.G_FALSE,
P_COMMIT IN VARCHAR2 DEFAULT CDR_PUB_DEF_CONSTANTS.G_FALSE,
P_VALIDATION_LEVEL IN NUMBER DEFAULT CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
X_RETURN_STATUS OUT NOCOPY VARCHAR2,
X_MSG_COUNT OUT NOCOPY NUMBER,
X_MSG_DATA OUT NOCOPY VARCHAR2,
PIO_ADAPTERDOMAINNAMING IN OUT NOCOPY CDR_NAMING_VERSION_OBJ_TYPE
);

Parameters This API has standard parameters (see "Standard Parameters" on page 5) and the following parameters:

- PARAM P_API_VERSION (Mandatory). Enter the current version of the API you are calling. The API compares the version numbers of incoming calls to its current version number and returns an error if they are incompatible.
- PARAM P_INIT_MSG_LIST (Optional). Accept the default value (FND_APIG_FALSE) to ensure that this individual API does not initialize the message list upon completion. Pass FND_APIG_TRUE to override the default behavior.
- PARAM P_COMMIT (Optional). Accept the default value (FND_APIG_FALSE) to ensure that this individual API does not get committed upon completion. Pass FND_APIG_TRUE to override the default behavior.
- PARAM P_VALIDATION_LEVEL (Optional). Accept the default value to perform full validation. No other values are currently supported.
- PARAM X_RETURN_STATUS. This output parameter returns the end status of the API: (S) Success, (E) Error or (U) Unexpected Error.
- PARAM X_MSG_COUNT. This output parameter returns the count of error messages if the return status is other than Success.
- PARAM X_MSG_DATA. This output parameter returns the text of the error message, if the message count is 1. If there are more than one message, use cdr_pub_msg_pub.get to retrieve the messages.
- PARAM PIO_ADAPTERDOMAINNAMING (Mandatory). This is a parameter of table type CDR_NAMING_VERSION_OBJ_TYPE that contains object attributes. Enter values to identify the Adapter Domain and enter new values for the
attributes you want to modify. All attributes are required. For OBJECT_TYPE_RC enter $OBJTYPES$ADAPTERDOMAIN.

19.2.3 Create an Adapter Area

Use this API to create an Adapter Area.

**Name**  CDR_PUB_ATK_ADAPTER.CreateAdapterArea

**Signature**

```sql
PROCEDURE CREATEADAPTERAREA(
    P_API_VERSION  IN    NUMBER,
    P_INIT_MSG_LIST  IN    VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_COMMIT  IN    VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_VALIDATION_LEVEL  IN    NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
    X_RETURN_STATUS  OUT    VARCHAR2,
    X_MSG_COUNT  OUT    NUMBER,
    X_MSG_DATA  OUT    VARCHAR2,
    PIO_ADAPTERAREANAMING  IN OUT    CDR_NAMING_VERSION_OBJ_TYPE,
    PIO_ADAPTERAREAROW  IN OUT    CDR_ADAPTER_AREAS%ROWTYPE
);```

**Parameters**  This API has standard parameters (see “Standard Parameters” on page 5) and the following parameters:

- **PIO_ADAPTERAREANAMING** (Mandatory) This is a parameter of table type CDR_NAMING_VERSION_OBJ_TYPE that contains object attributes. Enter values for the adapter area that you are creating.
- **PIO_ADAPTERAREAROW** (Mandatory) This is a parameter of row type CDR_ADAPTER_AREAS table that contains object attributes. Enter values specific for the adapter area that you are creating. For OBJECT_TYPE_RC enter $OBJTYPES$ADAPTERAREA.

The following attributes are required: COMPANY_ID, NAME

19.2.4 Modify an Adapter Area

Use this API to modify an Adapter Area.

**Name**  CDR_PUB_ATK_ADAPTER.ModifyAdapterArea

**Signature**

```sql
PROCEDURE MODIFYADAPTERAREA(
    P_API_VERSION IN NUMBER,
    P_INIT_MSG_LIST IN VARCHAR2 DEFAULT CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_COMMIT IN VARCHAR2 DEFAULT CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_VALIDATION_LEVEL IN NUMBER DEFAULT CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
    X_RETURN_STATUS OUT NOCOPY VARCHAR2,
    X_MSG_COUNT OUT NOCOPY NUMBER,
    X_MSG_DATA OUT NOCOPY VARCHAR2,
    PIO_ADAPTERAREANAMING IN OUT NOCOPY CDR_NAMING_VERSION_OBJ_TYPE,
    PIO_ADAPTERAREAROW IN OUT NOCOPY CDR_ADAPTER_AREAS%ROWTYPE
);```

**Parameters**  This API has standard parameters (see "Standard Parameters" on page 5) and the following parameters:
Define and Modify Adapters

- **PARAM P_INIT_MSG_LIST (Optional).** Accept the default value (FND_API.G_FALSE) to ensure that this individual API does not initialize the message list upon completion. Pass FND_API.G_TRUE to override the default behavior.

- **PARAM P_COMMIT (Optional).** Accept the default value (FND_API.G_FALSE) to ensure that this individual API does not get committed upon completion. Pass FND_API.G_TRUE to override the default behavior.

- **PARAM P_VALIDATION_LEVEL (Optional).** Accept the default value to perform full validation. No other values are currently supported.

- **PARAM X_RETURN_STATUS.** This output parameter returns the end status of the API: (S) Success, (E) Error or (U) Unexpected Error.

- **PARAM X_MSG_COUNT.** This output parameter returns the count of error messages if the return status is other than Success.

- **PARAM X_MSG_DATA.** This output parameter returns the text of the error message, if the message count is 1. If there are more than one message, use cdr_pub_msg_pub.get to retrieve the messages.

- **PARAM PIO_ADAPTERDOMAINNAMING (Mandatory).** This is a parameter of table-type CDR_NAMING_VERSION_OBJ_TYPE that contains object attributes. Enter values to identify the Adapter Area and enter new values for the attributes you want to modify.

  All attributes are required.

- **PARAM PIO_ADAPTERAREAROW (Mandatory).** This is a parameter of row-type CDR_ADAPTER_AREAS table that contains object attributes.

- **PARAM P_API_VERSION (Mandatory).** Enter the current version of the API you are calling. The API compares the version numbers of incoming calls to its current version number and returns an error if they are incompatible.

  For OBJECT_TYPE_RC enter $OBJTYPES$ADAPTERAREA.

19.2.5 Populate a Tech Type Table

Use this API to populate a Tech Type Table.

**Name**  CDR_PUB_ATK_ADAPTER.PopulateTechTypes

**Signature**

```sql
PROCEDURE POPULATETECHTYPES(
    P_API_VERSION  IN    NUMBER,
    P_INIT_MSG_LIST  IN    VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_COMMIT  IN    VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_VALIDATION_LEVEL  IN    NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
    X_RETURN_STATUS  OUT    VARCHAR2,
    X_MSG_COUNT  OUT    NUMBER,
    X_MSG_DATA  OUT    VARCHAR2,
    PIO_TECHTYPEROW  IN OUT    CDR_TECH_TYPES%ROWTYPE
);
```

**Parameters**  This API has standard parameters (see “Standard Parameters” on page 5) and the following parameter:

- **PIO_TECHTYPEROW (Mandatory)** This is a parameter of row type cdr_tech_types table that contains object attributes. Enter values specific for the tech type that you are creating.
19.2.6 Modify a Tech Type Table

Use this API to modify a Tech Type Table.

**Name**  
CDR_PUB_ATK_ADAPTER.ModifyTechTypes

**Signature**

```sql
PROCEDURE MODIFYTECHTYPE(
    P_API_VERSION IN NUMBER,
    P_INIT_MSG_LIST IN VARCHAR2 DEFAULT CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_COMMIT IN VARCHAR2 DEFAULT CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_VALIDATION_LEVEL IN NUMBER DEFAULT CDR_PUBL_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
    X_RETURN_STATUS OUT NOCOPY VARCHAR2,
    X_MSG_COUNT OUT NOCOPY NUMBER,
    X_MSG_DATA OUT NOCOPY VARCHAR2,
    PIO_TECHTYPEROW IN OUT NOCOPY CDR_TECH_TYPES%ROWTYPE,
);```

**Parameters**  
This API has standard parameters (see “Standard Parameters” on page 5) and the following parameters:

- **PARAM P_API_VERSION** (Mandatory). Enter the current version of the API you are calling. The API compares the version numbers of incoming calls to its current version number and returns an error if they are incompatible.

- **PARAM P_INIT_MSG_LIST** (Optional). Accept the default value (FND_API.G_FALSE) to ensure that this individual API does not initialize the message list upon completion. Pass FND_API.G_TRUE to override the default behavior.

- **PARAM P_COMMIT** (Optional). Accept the default value (FND_API.G_FALSE) to ensure that this individual API does not get committed upon completion. Pass FND_API.G_TRUE to override the default behavior.

- **PARAM P_VALIDATION_LEVEL** (Optional). Accept the default value to perform full validation. No other values are currently supported.

- **PARAM X_RETURN_STATUS**. This output parameter returns the end status of the API: (S) Success, (E) Error or (U) Unexpected Error.

- **PARAM X_MSG_COUNT**. This output parameter returns the count of error messages if the return status is other than Success.

- **PARAM X_MSG_DATA**. This output parameter returns the text of the error message, if the message count is 1. If there are more than one message, use cdr_pub_msg_pub.get to retrieve the messages.

- **PARAM PIO_TECHTYPEROW** (Mandatory). This is a parameter of row type cdr_tech_types table that contains object attributes. Enter values specific for the tech type that you want to update.

19.3 Host Definition Constants

This is a public interface that hosts definition constants for Oracle LSH APIs.

**Name**  
CDR_PUB_DEF_CONSTANTS

19.4 Get Factory Support

This is a public interface that hosts utility APIs for other packages.
This section contains the following topics:

- Section 19.4.1, "Get a Naming Version Object"
- Section 19.4.2, "Get a User ID"
- Section 19.4.3, "Get a User Name"

19.4.1 Get a Naming Version Object

Use this API to retrieve a valid CDR_NAMING_VERSION_OBJ_TYPE parameter by passing the primary keys COMPANY_ID, OBJ_ID, and OBJ_VER to it.

**Name**  
CDR_PUB_DEF_FACTORY_SUPPORT.GetNamingObject

**Signature**

FUNCTION GETNAMINGOBJECT(  
  P_API_VERSION IN NUMBER,  
  P_INIT_MSG_LIST IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,  
  P_COMMIT IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,  
  P_VALIDATION_LEVEL IN NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,  
  PI_NCOMPANYID IN CDR_NAMINGS.COMPANY_ID%TYPE,  
  PI_NOBJID IN CDR_NAMINGS.OBJ_ID%TYPE,  
  PI_NOBJVER IN CDR_NAMING_VERSIONS.OBJ_VER%TYPE,  
  PO_CDRNAMING OUT CDR_NAMING_VERSION_OBJ_TYPE  
) RETURN BOOLEAN;

**Return**  
BOOLEAN

**Parameters**  
This API has standard parameters (see "Standard Parameters" on page 5) and the following parameters:

- **PI_NCOMPANYID** (Mandatory) Enter the COMPANY_ID of the object.
- **PI_NOBJID** (Mandatory) Enter the OBJ_ID of the object.
- **PI_NOBJVER** (Mandatory) Enter the OBJ_VER of the object.
- **PO_CDRNAMING** (Mandatory) This is the output from the API.

19.4.2 Get a User ID

Use this API to retrieve a user ID.

**Name**  
CDR_PUB_DEF_FACTORY_SUPPORT.GetUserID

**Signature**

FUNCTION GETUSERID(  
  P_API_VERSION IN NUMBER,  
  P_INIT_MSG_LIST IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,  
  P_COMMIT IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,  
  P_VALIDATION_LEVEL IN NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL  
) RETURN CDR_DF_NAMING_V.CHECKED_OUT_ID%TYPE;

**Return**  
CDR_DF_NAMING_V.CHECKED_OUT_ID%TYPE

**Boolean**

**Parameters**  
This API has standard parameters. See "Standard Parameters" on page 5 for details.
19.4.3 Get a User Name

Use this API to retrieve the current user name.

**Name**  CDR_PUB_DEF_FACTORY_SUPPORT.GetUserName

**Signature**

FUNCTION GETUSERNAME(
  P_API_VERSION IN NUMBER,
  P_INIT_MSG_LIST IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_COMMIT IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_VALIDATION_LEVEL IN NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
  PI_USERID IN NUMBER
) RETURN VARCHAR2;

**Return**  Type VARCHAR2

**Description**  VARCHAR2.

**Parameters**  This API has standard parameters (see "Standard Parameters" on page 5) and the following parameter:

**PI_USERID**  (Mandatory) Enter the User ID. Use CDR_PUB_DEF_FACTORY_SUPPORT.GETUSERID to get the ID of the current user.

19.5 Get Factory Utilities

This is a public interface that hosts utility APIs for other packages.

This section contains the following topics:

- Section 19.5.1, "Get a Base Object Type"
- Section 19.5.2, "Get a Company ID"

19.5.1 Get a Base Object Type

Use this API to retrieve all details of an object from CDR_BASE_OBJ_TYPE table, by passing the object’s primary key values: COMPANY_ID, OBJ_ID, and OBJ_VER.

**Name**  CDR_PUB_DEF_FACTORY_UTILS.GetCDRBaseObject

**Signature**

FUNCTION GETCDRBASEOBJECT(
  P_API_VERSION IN NUMBER,
  P_INIT_MSG_LIST IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_COMMIT IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_VALIDATION_LEVEL IN NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
  PI_COMPANYID IN CDR_NAMINGS.COMPANY_ID%TYPE,
  PI_OBJID IN CDR_NAMINGS.OBJ_ID%TYPE,
  PI_OBJVER IN CDR_NAMING_VERSIONS.OBJ_VER%TYPE
) RETURN CDR_BASE_OBJ_TYPE;

**Return**  Type CDR_BASE_OBJ_TYPE

**Description**  CDR_BASE_OBJ_TYPE

**Parameters**  This API has standard parameters (see "Standard Parameters" on page 5) and the following parameters:
19.5.2 Get a Company ID

Use this API to retrieve the COMPANY_ID of a given object. The object is identified from the context that this API is used in.

**Name**  CDR_PUB_DEF_FACTORY_UTILS.GetCompanyID

**Signature**

FUNCTION GETCOMPANYID(
    P_API_VERSION  IN    NUMBER,
    P_INIT_MSG_LIST  IN    VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_COMMIT  IN    VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_VALIDATION_LEVEL  IN    NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL
) RETURN NUMBER;

**Return**  Type NUMBER

Description number Returns Company ID from the context

**Parameters**  This API has standard parameters. See “Standard Parameters” on page 5 for details.

19.6 Get Factory Validations

This is a public interface that hosts APIs for checking object validation on various objects. These APIs are tools to automatically validate objects without having to manually set their attributes to validate them.

This section contains the following topics:

-  Section 19.6.1, "Validate a Namespace"
-  Section 19.6.2, "Validate a Reference"

19.6.1 Validate a Namespace

Use this API to validate whether a given object is created in a valid parent; for example, you may want to check if a Program Definition is a valid parent of a Table Descriptor.

**Name**  CDR_PUB_DEF_FACTORY_VALIDATE.ValidateNamespace

**Signature**

FUNCTION VALIDATENAMESPACE(
    P_API_VERSION  IN    NUMBER,
    P_INIT_MSG_LIST  IN    VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_COMMIT  IN    VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_VALIDATION_LEVEL  IN    NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
    PIO_SOURCECDRNAMING  IN OUT    CDR_NAMING_VERSION_OBJ_TYPE
) RETURN BOOLEAN;

**Return**  Type BOOLEAN
Description boolean

**Parameters**  This API has standard parameters (see "Standard Parameters" on page 5) and the following parameter:

**PIO_SOURCECDRNAMING**  (Mandatory) This is a parameter of table type CDR_NAMING_VERSIONS_OBJ_TYPE that contains CDR Naming Version attributes.

For OBJECT_TYPE_RC enter the appropriate value for the object for which you want to validate a reference.

Other required attributes are: COMPANY_ID,OBJ_ID,OBJ_VER,OBJECT_TYPE_RC,NAMESPACE_OBJ_ID,NAMESPACE_OBJ_VER,REF_COMPANY_ID,REF_OBJ_ID,REF_OBJ_VER

### 19.6.2 Validate a Reference

Use this API to validate the definition specified for the given object id.

**Name**  CDR_PUB_DEF_FACTORY_VALIDATE.ValidateReference

**Signature**

```sql
FUNCTION VALIDATEREFERENCE(
  P_API_VERSION  IN    NUMBER,
  P_INIT_MSG_LIST  IN    VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_COMMIT  IN    VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_VALIDATION_LEVEL  IN    NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
  PIO_SOURCECDRNAMING  IN OUT    CDR_NAMING_VERSION_OBJ_TYPE
) RETURN BOOLEAN;
```

**Return**  Type BOOLEAN

If the function returns True, then the reference is valid, else it’s invalid.

**Parameters**  This API has standard parameters (see "Standard Parameters" on page 5) and the following parameter:

**PIO_SOURCECDRNAMING**  (Mandatory) This is a parameter of table type CDR_NAMING_VERSIONS_OBJ_TYPE that contains CDR Naming Version attributes.

For OBJECT_TYPE_RC enter the appropriate value for the object for which you want to validate a reference.

Other required attributes are: COMPANY_ID,OBJ_ID,OBJ_VER,OBJECT_TYPE_RC,NAMESPACE_OBJ_ID,NAMESPACE_OBJ_VER,REF_COMPANY_ID,REF_OBJ_ID,REF_OBJ_VER

### 19.7 Get Data from Naming Tables

This is a public interface that is used to retrieve data from naming-related tables.

This section contains the following topics:

- Section 19.7.1, "Get the Latest Version"
- Section 19.7.2, "Get a Maximum Version"
- Section 19.7.3, "Get the Type of a Naming Object"
- Section 19.7.4, "Get an Object’s Naming Version"
- Section 19.7.5, "Get an Object’s Subtype ID"
Get the Latest Version

Use this API to retrieve the latest version available for a given object.

**Name**  CDR_PUB_DF_NAMING_UTIL.GetLastVersion

**Signature**

FUNCTION GETLASTVERSION(
    P_API_VERSION IN NUMBER,
    P_INIT_MSG_LIST IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_COMMIT IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_VALIDATION_LEVEL IN NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
    PI_BASEOBJECT IN CDR_BASE_OBJ_TYPE
) RETURN NUMBER;

**Return**

Type NUMBER

Description number

**Parameters**  This API has standard parameters (see "Standard Parameters" on page 5) and the following parameter:

PI_BASEOBJECT (Mandatory)  This is a parameter of table type CDR_BASE_OBJ_TYPE that contains object attributes.

The required attributes are COMPANY_ID, OBJECT_ID, OBJECT_VER, NAMESPACE_ID, NAMESPACE_VER.

Get a Maximum Version

Use this API to get the maximum versions of a specified object and its namespace.

**Name**  CDR_PUB_DF_NAMING_UTIL.GetMaxObjAndNsVersions

**Signature**
PROCEDURE GETMAXOBJANDNSVERSIONS(
  P_API_VERSION IN NUMBER,
  P_INIT_MSG_LIST IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_COMMIT IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_VALIDATION_LEVEL IN NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
  X_RETURN_STATUS OUT VARCHAR2,
  X_MSG_COUNT OUT NUMBER,
  X_MSG_DATA OUT VARCHAR2,
  PI_COMPID IN NUMBER,
  PI_OBJID IN NUMBER,
  PO_MAXOBJVER OUT NUMBER,
  PO_MAXNSOBJVER OUT NUMBER
);

Parameters
This API has standard parameters (see "Standard Parameters" on page 5) and the following parameters:

- **PI_COMPID** (Mandatory) Enter the COMPANY_ID of the given object.
- **PI_OBJID** (Mandatory) Enter the OBJ_ID of the given object.
- **PO_MAXOBJVER** This is an output parameter. It returns the maximum version of the given object.
- **PO_MAXNSOBJVER** This is an output parameter. It returns the maximum version of the Namespace of the given object.

19.7.3 Get the Type of a Naming Object

Use this API to get the type of a specified naming object.

**Name**  CDR_PUB_DF_NAMING_UTIL.GetNamingObjectType

**Signature**

FUNCTION GETNAMINGOBJECTTYPE(
  P_API_VERSION IN NUMBER,
  P_INIT_MSG_LIST IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_COMMIT IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_VALIDATION_LEVEL IN NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
  PIO_BASEOBJECT IN CDR_BASE_OBJ_TYPE
) RETURN VARCHAR2;

Parameters
This API has standard parameters (see "Standard Parameters" on page 5) and the following parameter:

**PIO_BASEOBJECT** (Mandatory) This is a parameter of table type CDR_BASE_OBJ_TYPE that contains object attributes. The required attributes are COMPANY_ID, OBJECT_ID, OBJECT_VER, NAMESPACE_OBJ_ID, NAMESPACE_OBJ_VER.

19.7.4 Get an Object’s Naming Version

Use this API to retrieve an initialized object of table type CDR_NAMING_VERSION_OBJ_TYPE for a given naming object.

**Name**  CDR_PUB_DF_NAMING_UTIL.GetNamingVersionObject

**Signature**

FUNCTION GETNAMINGVERSIONOBJECT(
  P_API_VERSION IN NUMBER,
Get Data from Naming Tables

19.7.5 Get an Object's Subtype ID

Use this API to retrieve the object subtype ID of a given naming object.

**Name**  CDR_PUB_DF_NAMING_UTIL.GetObjectSubtypeID

**Signature**

FUNCTION GETOBJECTSUBTYPEID(
    P_API_VERSION  IN    NUMBER,
    P_INIT_MSG_LIST  IN    VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_COMMIT  IN    VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_VALIDATION_LEVEL  IN    NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
    PI_BASEOBJECT  IN    CDR_BASE_OBJ_TYPE
) RETURN NUMBER;

**Return**

Type NUMBER

Description Number

**Parameters**  This API has standard parameters (see "Standard Parameters" on page 5) and the following parameter:

PI_BASEOBJECT  (Mandatory) This is a parameter of table type CDR_BASE_OBJ_TYPE that contains object attributes.

The required attributes are COMPANY_ID, OBJECT_ID, OBJECT_VER, NAMESPACE_OBJ_ID, NAMESPACE_OBJ_VER.

19.7.6 Get an Object's Checkout Status

Use this API to find out whether or not a naming object is checked out.

**Name**  CDR_PUB_DF_NAMING_UTIL.IsNamingCheckedOut

**Signature**

FUNCTION ISNAMINGCHECKEDOUT(
    P_API_VERSION  IN    NUMBER,
)

**Return**

Type NUMBER

Description Number

**Parameters**  This API has standard parameters (see "Standard Parameters" on page 5) and the following parameter:

PI_BASEOBJECT  Mandatory) This is a parameter of table type CDR_BASE_OBJ_TYPE that contains object attributes.

The required attributes are COMPANY_ID, OBJECT_ID, OBJECT_VER, NAMESPACE_OBJ_ID, NAMESPACE_OBJ_VER.
Get Data from Naming Tables

P_INIT_MSG_LIST IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
P_COMMIT IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
P_VALIDATION_LEVEL IN NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
PIO_BASEOBJECT IN CDR_BASE_OBJ_TYPE
)
RETURN BOOLEAN;

Return
Type BOOLEAN
Description Boolean

Parameters This API has standard parameters (see "Standard Parameters" on page 5) and the following parameter:

PIO_BASEOBJECT (Mandatory) This is a parameter of table type CDR_BASE_OBJ_TYPE that contains object attributes.
The required attributes are COMPANY_ID, OBJECT_ID, OBJECT_VER, NAMESPACE_OBJ_ID, NAMESPACE_OBJ_VER.

19.7.7 Get Checkout Properties
Use this API to retrieve the checkout status and implicit checkout property of a specified object.

Name CDR_PUB_DF_NAMING_UTIL.GetChkoutProp

Signature
PROCEDURE GETCHKOUTPROP(
    P_API_VERSION IN NUMBER,
P_INIT_MSG_LIST IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
P_COMMIT IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
P_VALIDATION_LEVEL IN NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
    X_RETURN_STATUS OUT VARCHAR2,
    X_MSG_COUNT OUT NUMBER,
    X_MSG_DATA OUT VARCHAR2,
    PI_COMPID IN NUMBER,
    PI_OBJID IN NUMBER,
    PO_CHKOUTSTATUS OUT VARCHAR2,
    PO_CHKIMPLPROP OUT VARCHAR2
);

Parameters This API has standard parameters (see "Standard Parameters" on page 5) and the following parameters:

- PI_COMPID (Mandatory) Enter the COMPANY_ID of the given object.
- PI_OBJID (Mandatory) Enter the OBJ_ID of the given object.
- PO_CHKOUTSTATUS This is an output parameter. It returns the checkout status of the given object.
- PO_CHKIMPLPROP This is an output parameter. It returns the implicit checkout property of the given object.

19.7.8 Get a Naming Object's Parent
Use this API to retrieve the parent object of a given naming object.
Get Data from Naming Tables

Name  CDR_PUB_DF_NAMING_UTIL.GetParentNaming

Signature
FUNCTION GETPARENTNAMING(
  P_API_VERSION IN    NUMBER,
  P_INIT_MSG_LIST IN    VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_COMMIT IN    VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_VALIDATION_LEVEL IN    NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
  PI_BASEOBJECT IN    CDR_BASE_OBJ_TYPE
) RETURN CDR_BASE_OBJ_TYPE;

Return
Type CDR_BASE_OBJ_TYPE
Description CDR_BASE_OBJ_TYPE

Parameters  This API has standard parameters (see "Standard Parameters" on page 5) and the following parameter:

PI_BASEOBJECT (Mandatory) This is a parameter of table type CDR_BASE_OBJ_TYPE that contains object attributes.

The required attributes are COMPANY_ID,OBJECT_ID,OBJECT_VER,NAMESPACE_OBJ_ID,NAMESPACE_OBJ_VER.

19.7.9 Get a Parent Naming Object

Use this API to retrieve the parent object of a naming object.

Name  CDR_PUB_DF_NAMING_UTIL.GetParentNaming

Signature
FUNCTION GETPARENTNAMING(
  P_API_VERSION IN    NUMBER,
  P_INIT_MSG_LIST IN    VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_COMMIT IN    VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_VALIDATION_LEVEL IN    NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
  PI_BASEOBJECT IN    CDR_BASE_OBJ_TYPE
) RETURN CDR_BASE_OBJ_TYPE;

Return
Type CDR_BASE_OBJ_TYPE
Description CDR_BASE_OBJ_TYPE

Parameters  This API has standard parameters (see "Standard Parameters" on page 5) and the following parameter:

PI_BASEOBJECT (Mandatory) This is a parameter of table type CDR_BASE_OBJ_TYPE that contains object attributes.

The required attributes are COMPANY_ID,OBJECT_ID,OBJECT_VER,NAMESPACE_OBJ_ID,NAMESPACE_OBJ_VER.

19.7.10 Get the Latest Version of the Parent Object

Use this API to retrieve the latest version of the parent of the given object.
Get Data from Naming Tables

**Name**  CDR_PUB_DF_NAMING_UTIL.GetLatestVersionOfParent

**Signature**

FUNCTION GETLATESTVERSIONOFPARENT(
  P_API_VERSION  IN    NUMBER,
  P_INIT_MSG_LIST  IN    VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_COMMIT  IN    VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_VALIDATION_LEVEL  IN    NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
  PI_BASEOBJECT  IN    CDR_BASE_OBJ_TYPE
) RETURN CDR_BASE_OBJ_TYPE;

**Return**
Type CDR_BASE_OBJ_TYPE
Description cdr_base_obj_type

**Parameters**  This API has standard parameters (see "Standard Parameters" on page 5) and the following parameter:

- **PI_BASEOBJECT** (Mandatory) This is a parameter of table type CDR_BASE_OBJ_TYPE that contains object attributes.
  The required attributes are COMPANY_ID, OBJECT_ID, OBJECT_VER, NAMESPACE_OBJ_ID, NAMESPACE_OBJ_VER.

19.7.11  Get the Naming Status of a Parent Object

Use this API to get the naming status of the parent object of the specified object.

**Name**  CDR_PUB_DF_NAMING_UTIL.GetParentNamingStatus

**Signature**

FUNCTION GETPARENTNAMINGSTATUS(
  P_API_VERSION  IN    NUMBER,
  P_INIT_MSG_LIST  IN    VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_COMMIT  IN    VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_VALIDATION_LEVEL  IN    NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
  PI_COMPANYID  IN    CDR_NAMINGS.COMPANY_ID%TYPE,
  PI_NSOBJID  IN    CDR_NAMINGS.OBJ_ID%TYPE,
  PI_NSOBJVER  IN    CDR_NAMING_VERSIONS.OBJ_VER%TYPE
) RETURN VARCHAR2;

**Return**
Type VARCHAR2
Description varchar2

**Parameters**  This API has standard parameters (see "Standard Parameters" on page 5) and the following parameters:

- **PI_COMPANYID** (Mandatory) Enter the COMPANY_ID of the object.
- **PI_NSOBJJID** (Mandatory) Enter the OBJ_ID of the given object.
- **PI_NSOBJVER** (Mandatory) Enter the OBJ_VER of the given object.
19.7.12 Get the Validation Status of a Parent Object

Use this API to get the validation status of the parent object of the specified object.

**Name**  
CDR_PUB_DF_NAMING_UTIL.GetParentValidationStatus

**Signature**

```sql
FUNCTION GETPARENTVALIDATIONSTATUS(
    P_API_VERSION  IN    NUMBER,
    P_INIT_MSG_LIST  IN    VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_COMMIT  IN    VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_VALIDATION_LEVEL  IN    NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
    PI_COMPANYID  IN    CDR_NAMINGS.COMPANY_ID%TYPE,
    PI_NSOBJID  IN    CDR_NAMINGS.OBJ_ID%TYPE,
    PI_NSOBJVER  IN    CDR_NAMING_VERSIONS.OBJ_VER%TYPE
) RETURN VARCHAR2;
```

**Return**

Type VARCHAR2

**Description** varchar2

**Parameters**

This API has standard parameters (see "Standard Parameters" on page 5) and the following parameters:

- **PI_COMPANYID** (Mandatory) This refers to the company id of the given object.
- **PI_NSOBJID** (Mandatory) Enter the OBJ_ID of the given object.
- **PI_NSOBJVER** (Mandatory) Enter the OBJ_VER of the given object.

19.7.13 Get a Definition Object

Use this API to retrieve the definition object that the given instance object points to.

**Name**  
CDR_PUB_DF_NAMING_UTIL.GetRefNaming

**Signature**

```sql
FUNCTION GETREFNAMING(
    P_API_VERSION  IN    NUMBER,
    P_INIT_MSG_LIST  IN    VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_COMMIT  IN    VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_VALIDATION_LEVEL  IN    NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
    PI_BASEOBJECT  IN    CDR_BASE_OBJ_TYPE
) RETURN CDR_BASE_OBJ_TYPE;
```

**Return**

Type CDR_BASE_OBJ_TYPE

**Description** CDR_BASE_OBJ_TYPE

**Parameters**

This API has standard parameters (see "Standard Parameters" on page 5) and the following parameter:

- **PI_BASEOBJECT** (Mandatory) This is a parameter of table type CDR_BASE_OBJ_TYPE that contains object attributes.

  The required attributes are COMPANY_ID,OBJECT_ID,OBJECT_VER,NAMESPACE_OBJ_ID,NAMESPACE_OBJ_VER.
19.7.14 Get a Lookup Meaning

Use this API to get the Lookup Meaning for a Lookup Type and Code

**Name**  CDR\_PUB\_DF\_NAMING\_UTIL\_GetLookupMeaning

**Signature**

PROCEDURE GETLOOKUPMEANING(
    P\_API\_VERSION IN NUMBER,
    P\_INIT\_MSG\_LIST IN VARCHAR2 := CDR\_PUB\_DEF\_CONSTANTS.G\_FALSE,
    P\_COMMIT IN VARCHAR2 := CDR\_PUB\_DEF\_CONSTANTS.G\_FALSE,
    P\_VALIDATION\_LEVEL IN NUMBER := CDR\_PUB\_DEF\_CONSTANTS.G\_VALID\_LEVEL\_FULL,
    X\_RETURN\_STATUS OUT VARCHAR2,
    X\_MSG\_COUNT OUT NUMBER,
    X\_MSG\_DATA OUT VARCHAR2,
    PI\_LOOKUPTYPE IN VARCHAR2,
    PI\_LOOKUPCODE IN VARCHAR2,
    PO\_LOOKUPMEANING OUT VARCHAR2
);

**Parameters**  This API has standard parameters (see "Standard Parameters" on page 5) and the following parameters:

- **PI\_LOOKUPTYPE** (Mandatory) Enter the Lookup Type
- **PI\_LOOKUPCODE** (Mandatory) Enter the Lookup Code
- **PO\_LOOKUPMEANING** This is an output parameter. It returns the Lookup Meaning.

19.7.15 Find Whether an Object is an Instance

Use this API to check if the given naming object is an instance.

**Name**  CDR\_PUB\_DF\_NAMING\_UTIL\_IsInstance

**Signature**

FUNCTION ISINSTANCE(
    P\_API\_VERSION IN NUMBER,
    P\_INIT\_MSG\_LIST IN VARCHAR2 := CDR\_PUB\_DEF\_CONSTANTS.G\_FALSE,
    P\_COMMIT IN VARCHAR2 := CDR\_PUB\_DEF\_CONSTANTS.G\_FALSE,
    P\_VALIDATION\_LEVEL IN NUMBER := CDR\_PUB\_DEF\_CONSTANTS.G\_VALID\_LEVEL\_FULL,
    PI\_OBJECT\_TYPE\_RC IN VARCHAR2
) RETURN VARCHAR2;

**Return**

Type VARCHAR2

Description varchar2

**Parameters**  This API has standard parameters (see "Standard Parameters" on page 5) and the following parameter:

**PI\_OBJECT\_TYPE\_RC** (Mandatory) Object Type RC.
19.7.16 Find Whether Checked Out By Current User

Use this API to find out whether the given naming object is checked out by the current user.

**Name**  CDR_PUB_DF_NAMING_UTIL.IsCheckedOutByUser

**Signature**

FUNCTION ISCHECKEDOUTBYUSER(
  P_API_VERSION IN    NUMBER,
  P_INIT_MSG_LIST  IN    VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_COMMIT  IN    VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_VALIDATION_LEVEL  IN    NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
  PIO_BASEOBJECT  IN    CDR_BASE_OBJ_TYPE
) RETURN BOOLEAN;

**Return**

Type BOOLEAN

Description boolean

**Parameters**  This API has standard parameters (see "Standard Parameters" on page 5) and the following parameter:

PIO_BASEOBJECT (Mandatory) This is a parameter of table type CDR_BASE_OBJ_TYPE that contains object attributes.

The required attributes are COMPANY_ID,OBJECT_ID,OBJECT_VER,NAMESPACE_OBJ_ID,NAMESPACE_OBJ_VER.

19.7.17 Find Whether a Checkout is User-Specific

Use this API to find out if an object is user specific or non-user specific. Only the user who checked out a user specific object can check it in; whereas any user may check in a non-user specific object.

**Name**  CDR_PUB_DF_NAMING_UTIL.IsCheckOutUserSpecific

**Signature**

FUNCTION ISCHECKOUTUSERSPECIFIC(
  P_API_VERSION IN    NUMBER,
  P_INIT_MSG_LIST  IN    VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_COMMIT  IN    VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_VALIDATION_LEVEL  IN    NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
  PI_SOURCECDRNAMING  IN    CDR_NAMING_VERSION_OBJ_TYPE
) RETURN BOOLEAN;

**Return**

Type BOOLEAN

Description Boolean True or False

**Parameters**  This API has standard parameters (see "Standard Parameters" on page 5) and the following parameter:

PI_SOURCECDRNAMING (Mandatory) This is a parameter of type cdr_naming_versions_obj_type.
The following attributes are required: COMPANY_ID, OBJECT_TYPE_RC, NAME, NAMESPACE_OBJ_ID, NAMESPACE_OBJ_VER, OWNING_LOCATION_RC, OBJECT_SUBTYPE_ID, DESCRIPTION, REF_COMPANY_ID, REF_OBJ_ID, REF_OBJ_VER.

19.7.18 Find Whether Checkout is Implicit

Use this API to find out whether a naming object is checked out implicitly.

**Name**  CDR_PUB_DF_NAMING_UTIL.IsCheckOutImplicit

**Signature**

FUNCTION ISCHECKOUTIMPLICIT(
    P_API_VERSION IN NUMBER,
    P_INIT_MSG_LIST IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_COMMIT IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_VALIDATION_LEVEL IN NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
    PI_OBJECT_TYPE_RC IN VARCHAR2,
    PI_ISINSTONLY IN VARCHAR2
) RETURN BOOLEAN;

**Return**

Type BOOLEAN

Description boolean

**Parameters** This API has standard parameters (see "Standard Parameters" on page 5) and the following parameters:

- **PI_OBJECT_TYPE_RC** (Mandatory) Enter the Object Type.
- **PI_ISINSTONLY** (Mandatory) This will have the value 'YES' or 'NO', to indicate whether it is called from an Instance or a Definition.

19.8 Read Messages

This is a public interface for reporting using messages from the system’s message stack. See "Code Example Using Security and Error Message APIs" on page 1-5 for an example of a program that calls this reporting API.

This section contains the following topics:

- Section 19.8.1, "Get a Message"
- Section 19.8.2, "Get a Message Count"
- Section 19.8.3, "Initialize a Message Stack"

19.8.1 Get a Message

Use this API to retrieve messages from the message stack.

**Name**  CDR_PUB_MSG_PUB.Get

**Signature**

FUNCTION GET(
    P_API_VERSION IN NUMBER,
    P_INIT_MSG_LIST IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
P_COMMIT IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
P_VALIDATION_LEVEL IN NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
P_MSG_INDEX IN NUMBER := G_NEXT,
P_ENCODED IN VARCHAR2 := 'T'
) RETURN VARCHAR2;

Return
Type VARCHAR2
Description varchar2 Message Text

Parameters This API has standard parameters (see "Standard Parameters" on page 5) and the following parameters:

- **P_MSG_INDEX** (Mandatory) Enter the number of the message you want to retrieve; for example, if you enter the number 1, the first message is retrieved from the message stack.
- **P_ENCODED** (Mandatory) Enter "T" if you want the message to be encoded, and "F" if you do not.

19.8.2 Get a Message Count

Use this API to retrieve the count of messages in the message stack. This API returns the G_MSG_COUNT value.

Name CDR_PUB_MSG_PUB.Count_Msg

Signature

FUNCTION COUNT_MSG(
    P_API_VERSION IN NUMBER,
    P_INIT_MSG_LIST IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_COMMIT IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_VALIDATION_LEVEL IN NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL
) RETURN NUMBER;

Return
Type NUMBER
Description Number of messages in the stack

Parameters This API has standard parameters (see "Standard Parameters" on page 5).

19.8.3 Initialize a Message Stack

Use this API to initialize the global message table. This API clears the G_MSG_TBL and resets all its global variables, except the message level threshold value.

Name CDR_PUB_MSG_PUB.Initialize

Signature

PROCEDURE INITIALIZE(
    P_API_VERSION IN NUMBER,
    P_INIT_MSG_LIST IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_COMMIT IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_VALIDATION_LEVEL IN NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
) RETURN;
X_RETURN_STATUS OUT VARCHAR2,
X_MSG_COUNT OUT NUMBER,
X_MSG_DATA OUT VARCHAR2
);

Parameters  This API has standard parameters (see “Standard Parameters” on page 5).
This is a public interface for Execution Setup-related operations, including creating, modifying, and removing Execution Setups. It also includes functions for checking Execution Setups in and out.

20.1 Create and Modify Execution Setups

This section contains the following topics:

- Section 20.1.1, "Create an Execution Setup"
- Section 20.1.2, "Check Out an Execution Setup"
- Section 20.1.3, "Modify an Execution Setup"
- Section 20.1.4, "Modify a Parameter"
- Section 20.1.5, "Modify an Execution Setup Parameter"
- Section 20.1.6, "Load Parameter Details"
- Section 20.1.7, "Copy an Execution Setup"
- Section 20.1.8, "Check In an Execution Setup"
- Section 20.1.9, "Submit an Execution Setup"
- Section 20.1.10, "Submit an Execution Setup for Instances"
- Section 20.1.11, "Submit an Execution Setup for Compound Objects"
- Section 20.1.12, "Upgrade an Execution Setup"
- Section 20.1.13, "Upgrade All Execution Setups"
- Section 20.1.14, "Make an Execution Setup Active"
- Section 20.1.15, "Remove an Execution Setup"

20.1.1 Create an Execution Setup

Use this API to create a new Execution Setup.

**Name**  
CDR_PUB_DF_EXECUTIONSETUP.CreateExecutionSetup

**Signature**

```sql
PROCEDURE CREATEEXECUTIONSETUP(  
P_API_VERSION IN NUMBER,  
P_INIT_MSG_LIST IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,  
P_COMMIT IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,  
```
Create and Modify Execution Setups

P_VALIDATE_LEVEL IN NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
X_RETURN_STATUS OUT VARCHAR2,
X_MSG_COUNT OUT NUMBER,
X_MSG_DATA OUT VARCHAR2,
PIO_SOURCECDRNAMING IN OUT CDR_NAMING_VERSION_OBJ_TYPE,
PO_DEFCLASSIFICATIONCOLL IN CDR_CLASSIFICATIONS_COLL
);

Parameters  This API has standard parameters (see "Standard Parameters" on page 5) and the following parameters:

■ PIO_SOURCECDRNAMING (Mandatory) This is a parameter of table type CDR_NAMING_VERSIONS_OBJ_TYPE that contains CDR Naming Version attributes. For OBJECT_TYPE_RC enter $OBJTYPES$EXECSETUP.

■ PO_DEFCLASSIFICATIONCOLL (Optional) By default the new definition is classified according to the subtype you assigned it in the CDR_NAMING_VERSION_OBJ_TYPE.

If you want to override the default classifications for one or more classification levels, use this parameter. This is a collection of CDR_CLA_OBJ_TYPES, which have 5 attributes, including CLA_LEVEL_ID and CLASSIFICATION_ID.

If you want the definition to inherit its classifications for a particular level from its parent, enter the classification level ID and, for the CLASSIFICATION_ID, enter 0 (zero).

If you want to explicitly assign one or more terms for a particular level, initialize a CDR_CLA_OBJ_TYPE for each term, entering the classification level ID and, for the CLASSIFICATION_ID, the term ID. The PAR_ attributes are not relevant to Execution Setups. Do not enter any values for them. If you are not creating a new definition, do not enter values here.

20.1.2 Check Out an Execution Setup

Use this API to check out an Execution Setup.

Name  CDR_PUB_DF_EXECUTIONSETUP.CheckOutExecutionSetup

Signature

PROCEDURE CHECKOUTEXECUTIONSETUP(
  P_API_VERSION IN NUMBER,
P_INIT_MSG_LIST IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
P_COMMIT IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
P_VALIDATE_LEVEL IN NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
X_RETURN_STATUS OUT VARCHAR2,
X_MSG_COUNT OUT NUMBER,
X_MSG_DATA OUT VARCHAR2,
PIO_BASEOBJECT IN OUT CDR_BASE_OBJ_TYPE,
PI_COMMENT IN VARCHAR2,
PI_ISINSTONLY IN VARCHAR2
);

Parameters  This API has standard parameters (see "Standard Parameters" on page 5) and the following parameters:

■ PIO_BASEOBJECT (Mandatory) This is a parameter of table type CDR_BASE_OBJ_TYPE that contains object attributes. Enter values to identify the Execution Setup that you want to check out.
The following attributes are required: COMPANY_ID, OBJ_ID, OBJ_VER, OBJECT_VERSION_NUMBER, NAMESPACE_OBJ_ID, NAMESPACE_OBJ_VER.

- **PI_COMMENT** (Optional) Enter the reason you are checking out the Execution Setup.
- **PI_ISINSTONLY** (Mandatory) Enter $YESNO$NO.

### 20.1.3 Modify an Execution Setup

Use this API to modify an Execution Setup. You can change an Execution Setup’s name and description. You must check out the Execution Setup before modifying it.

**Name**  CDR_PUB_DF_EXECUTIONSETUP.ModifyExecutionSetup

**Signature**

PROCEDURE MODIFYEXECUTIONSETUP(
  P_API_VERSION IN NUMBER,
  P_INIT_MSG_LIST IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_COMMIT IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_VALIDATION_LEVEL IN NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
  X_RETURN_STATUS OUT VARCHAR2,
  X_MSG_COUNT OUT NUMBER,
  X_MSG_DATA OUT VARCHAR2,
  PIO_SOURCECDRNAMING IN OUT CDR_NAMING_VERSION_OBJ_TYPE
);

**Parameters**  This API has standard parameters (see "Standard Parameters" on page 5) and the following parameter:

- **PIO_SOURCECDRNAMING** (Mandatory) This is a parameter of table type CDR_NAMING_VERSION_OBJ_TYPE that contains object attributes. Enter values to identify the Execution Setup and enter new values for the attributes you want to modify. All attributes are required.

---

**Note:**  Use separate APIs for modifying the validation status and the version label: CDR_PUB_VL_VALIDATION.UPDATEVALSTATUS and CDR_PUB_DF_NAMING.UPDATEVERSIONLABEL.

### 20.1.4 Modify a Parameter

Use this API to modify Execution Setup parameters.

**Name**  CDR_PUB_DF_EXECUTIONSETUP.ModifyParameter

**Signature**

PROCEDURE MODIFYPARAMETER(
  P_API_VERSION IN NUMBER,
  P_INIT_MSG_LIST IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_COMMIT IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_VALIDATION_LEVEL IN NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
  X_RETURN_STATUS OUT VARCHAR2,
  X_MSG_COUNT OUT NUMBER,
  X_MSG_DATA OUT VARCHAR2,
  PI_CHANGEDPARAMETERCOLL IN CDR_SUBMISSION_DETAILS_COLL
);

---

Execution Setups 20-3
Parameters This API has standard parameters (see "Standard Parameters" on page 5) and the following parameter:

PI_CHANGEDPARAMETERCOLL (Mandatory) This is a collection of CDR_SUBMISSION_DETAILS.Collection of Parameter Instances on which the modify operation has been requested.

20.1.5 Modify an Execution Setup Parameter

Use this API to modify the parameters of an Execution Setup.

Name CDR_PUB_DF_EXECUTIONSETUP.ModifyYesParameters

Signature

PROCEDURE MODIFYYESPARAMETERS(
    P_API_VERSION IN NUMBER,
    P_INIT_MSG_LIST IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_COMMIT IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_VALIDATION_LEVEL IN NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
    X_RETURN_STATUS OUT VARCHAR2,
    X_MSG_COUNT OUT NUMBER,
    X_MSG_DATA OUT VARCHAR2,
    PIO_PARAMETERCOLL IN OUT CDR_PARAMETER_COLL
);

Parameters This API has standard parameters (see "Standard Parameters" on page 5) and the following parameter:

PIO_PARAMETERCOLL (Mandatory) This is a collection of cdr_parameter_coll. Enter the values for parameters you want to change.

The attributes you can change are: read_only_flag_rc, mandatory_flag_rc, and visible_flag_rc. Enter $yesno$yes for yes, and $yesno$no for no.

20.1.6 Load Parameter Details

Use this API to insert all parameters into a CDR_ES_TEMP table before submitting Report Set instance and Work Flow instance jobs.

Name CDR_PUB_DF_EXECUTIONSETUP.LoadESTempTable

Signature

PROCEDURE LOADESTEMPTABLE(
    P_API_VERSION IN NUMBER,
    P_INIT_MSG_LIST IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_COMMIT IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_VALIDATION_LEVEL IN NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
    X_RETURN_STATUS OUT VARCHAR2,
    X_MSG_COUNT OUT NUMBER,
    X_MSG_DATA OUT VARCHAR2,
    PI_BASEOBJ IN CDR_BASE_OBJ_TYPE,
    PI_NODEPRREFID IN NUMBER := NULL,
    PI_NODEPRREFVER IN NUMBER := NULL,
    PI_JOBID IN NUMBER := NULL
);

Parameters This API has standard parameters (see "Standard Parameters" on page 5) and the following parameters:
Create and Modify Execution Setups

20.1.7 Copy an Execution Setup

Use this API to duplicate one or more Execution Setups for an object.

**Name**  
CDR_PUB_DF_EXECUTIONSETUP.CopyExecutionSetup

**Signature**

```sql
PROCEDURE COPYEXECUTIONSETUP(
    P_API_VERSION IN NUMBER,
    P_INIT_MSG_LIST IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_COMMIT IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_VALIDATION_LEVEL IN NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
    X_RETURN_STATUS OUT VARCHAR2,
    X_MSG_COUNT OUT NUMBER,
    X_MSG_DATA OUT VARCHAR2,
    PI_EXECUTIONSETUP IN OUT CDR_BASE_OBJ_COLL
);
```

**Parameters**  
This API has standard parameters (see “Standard Parameters” on page 5) and the following parameter:

- **PI_EXECUTIONSETUP**  
(Mandatory) This is a collection of CDR_BASE_OBJ_TYPES. For each Execution Setup that you want to copy, initialize a CDR_BASE_OBJ_TYPE and then extend the collection.

   The following attributes are required: COMPANY_ID, OBJ_ID, OBJ_VER, OBJECT_VERSION_NUMBER, NAMESPACE_OBJ_ID, NAMESPACE_OBJ_VER.

20.1.8 Check In an Execution Setup

Use this API to check in an Execution Setup.

**Name**  
CDR_PUB_DF_EXECUTIONSETUP.CheckInExecutionSetup

**Signature**

```sql
PROCEDURE CHECKINEXECUTIONSETUP(
    P_API_VERSION IN NUMBER,
    P_INIT_MSG_LIST IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_COMMIT IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_VALIDATION_LEVEL IN NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
    X_RETURN_STATUS OUT VARCHAR2,
    X_MSG_COUNT OUT NUMBER,
    X_MSG_DATA OUT VARCHAR2,
    PIO_BASEOBJECT IN OUT CDR_BASE_OBJ_TYPE,
    PI_COMMENT IN VARCHAR2
);
```
Parameters This API has standard parameters (see "Standard Parameters" on page 5) and the following parameters:

- **PIO_BASEOBJECT** (Mandatory) This is a parameter of table type CDR_BASE_OBJ_TYPE that contains object attributes. Enter values to identify the Execution Setup that you want to check in.
  
The following attributes are required: COMPANY_ID,OBJ_ID,OBJ_VER,OBJECT_VERSION_NUMBER,NAMESPACE_OBJ_ID,NAMESPACE_OBJ_VER

- **PI_COMMENT** (Optional) Enter the reason you are checking in the Execution Setup.

20.1.9 Submit an Execution Setup

Use this API to submit an Execution Setup.

Name CDR_PUB_DF_EXECUTIONSETUP.SubmitExecutionSetup

Signature

PROCEDURE SUBMITEXECUTIONSETUP(
  P_API_VERSION IN NUMBER,
  P_INIT_MSG_LIST IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_COMMIT IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_VALIDATION_LEVEL IN NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
  X_RETURN_STATUS OUT VARCHAR2,
  X_MSG_COUNT OUT NUMBER,
  X_MSG_DATA OUT VARCHAR2,
  PI_EXECUTIONSETUP IN CDR_BASE_OBJ_TYPE,
  PI_CHANGEDSYSTEMPARAMETERCOLL IN CDR_SUBMISSION_DETAILS_COLL,
  PI_TICOLL IN CDR_SNAPSHOT_TABLE_COLL,
  PIINCLUDEDOBJCOLL IN CDR_SUBMISSION_DETAILS_COLL,
  PI_CURRENTJOBID OUT VARCHAR2,
  PI_OUTPUTTITLE IN VARCHAR2,
  PI_OUTPUTDESC IN VARCHAR2
);

Parameters This API has standard parameters (see "Standard Parameters" on page 5) and the following parameters:

- **PI_EXECUTIONSETUP** (Mandatory) This is a parameter of table type CDR_BASE_OBJ_TYPE that contains object attributes. Enter values to identify the Execution Setup that you want to submit.
  
The following attributes are required: COMPANY_ID,OBJ_ID,OBJ_VER,OBJECT_VERSION_NUMBER,NAMESPACE_OBJ_ID,NAMESPACE_OBJ_VER.

- **PI_CHANGEDSYSTEMPARAMETERCOLL**

- **PI_TICOLL** (Mandatory) This is a collection of CDR_SNAPSHOT_TABLE_OBJ_TYPE.

- **PIINCLUDEDOBJCOLL** (Mandatory) This is a collection of CDR_SUBMISSION_DETAILS_OBJ_TYPE that contains PRREF_IDS of Programs to include in the Execution Setup. For Report Sets and Report Set Entries, the user can include only selected Report Sets for execution. Enter only those PRREF_IDS.

- **PI_CURRENTJOBID** (Mandatory) Enter the JOB_ID of the Job that this Execution Setup submission created. This parameter is used to generate user's feedback.
Create and Modify Execution Setups

Execution Setups

- **PI_OUTPUTTITLE** (Mandatory) Enter a title for the output from this Execution Setup.
- **PI_OUTPUTDESC** (Mandatory) Enter the description for the output from this Execution Setup.

### 20.1.10 Submit an Execution Setup for Instances

Use this API to submit an Execution Setup for Report Set instances and Workflow instances. You must call the loadESTempTable API before calling this API.

**Name**  
CDR_PUB_DF_EXECUTIONSETUP.SubmitExecutionSetup

**Signature**

```sql
PROCEDURE SUBMITEEXECUTIONSETUP(
    P_API_VERSION IN NUMBER,
    P_INIT_MSG_LIST IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_COMMIT IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_VALIDATION_LEVEL IN NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
    X_RETURN_STATUS OUT VARCHAR2,
    X_MSG_COUNT OUT NUMBER,
    X_MSG_DATA OUT VARCHAR2,
    PI_EXECUTIONSETUP IN OUT cdr_base_obj_type,
    PIO_CHANGEDSYSTEMPARAMETERCOLL IN OUT cdr_submission_details_coll,
    PI_TICOLL IN cdr_snapshot_table_coll,
    PO_CURRENTJOBID OUT VARCHAR2,
    PI_OUTPUTTITLE IN VARCHAR2,
    PI_OUTPUTDESC IN VARCHAR2
);
```

**Parameters**  
This API has standard parameters (see "Standard Parameters" on page 5) and the following parameters:

- **PI_EXECUTIONSETUP** (Mandatory) This is a parameter of cdr_base_obj_type.
  
Enter values for attributes that describe the Execution Setup: COMPANY_ID, OBJ_ID, OBJ_VER, NAMESPACE_OBJ_ID, NAMESPACE_OBJ_VER, OBJECT_VERSION_NUMBER.

- **PIO_CHANGEDSYSTEMPARAMETERCOLL** (Mandatory) This is a parameter of cdr_submission_details_coll type.
  
  You must provide values for the following system parameters: company_id (use CDR_DF_PUB_DEF_CONSTANTS.CURRENT_COMPANY_ID); submission_id (enter null); prref_id (enter -1), prref_ver (enter -1), parameter_ref_obj_id, and parameter_ref_obj_id. Query cdr_parameters on company_id, obj_id, and obj_ver to find values for the default_value attribute. Use this value to set the attribute parameter_value.

- **PI_TICOLL** (Mandatory) This is a collection of data snapshots. This value may be null if you want to use current data.
  
  Otherwise the required attributes are: TI_OBJ_ID, TI_OBJ_VER, SRC_MASTER_JOB_ID, SRC_COMPANY_ID

- **PO_CURRENTJOBID** (Mandatory) This is an output parameter. The system generates a Job ID for the submitted Execution Setup. You can print this parameter through your program.

- **PI_OUTPUTTITLE** (Mandatory) Enter a title for the Execution Setup. It is easy to track an Execution Setup with a meaningful title.
PI_OUTPUTDESC (Mandatory) Enter a description for the Execution Setup output.

20.1.11 Submit an Execution Setup for Compound Objects

Use this API to submit an Execution Setup for compound objects such as Report Sets and Workflows.

The API takes care of both, populating the temporary tables for execution and submitting the job, so that you do not need to invoke the loadESTempTable explicitly.

**Name**  CDR_PUB_DF_EXECUTIONSETUP.SubmitExecutionSetup

**Signature**

```sql
PROCEDURE SUBMITEXECUTIONSETUP(
  P_API_VERSION IN NUMBER,
  P_INIT_MSG_LIST IN VARCHAR2 DEFAULT CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_COMMIT IN VARCHAR2 DEFAULT CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_VALIDATION_LEVEL IN NUMBER DEFAULT CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
  X_RETURN_STATUS OUT NOCOPY VARCHAR2,
  X_MSG_COUNT OUT NOCOPY NUMBER,
  X_MSG_DATA OUT NOCOPY VARCHAR2,
  PI_EXECUTIONSETUP IN OUT NOCOPY CDR_BASE_OBJ_TYPE,
  PIO_CHANGEDSYSTEMPARAMETERCOLL IN OUT NOCOPY CDR_SUBMISSION_DETAILS_COLL,
  PI_TICOLL IN CDR_SNAPSHOT_TABLE_COLL,
  PI_EXEPARAMCOLL IN OUT NOCOPY CDR_PARAMETER_COLL,
  PI_EXEOBJCOLL IN OUT NOCOPY CDR_PARAMETER_COLL,
  PI_NODEPRREFID IN NUMBER DEFAULT NULL,
  PI_NODEPRREFVER IN NUMBER DEFAULT NULL,
  PI_JOBID IN NUMBER DEFAULT NULL,
  PO_JOBDID OUT NOCOPY VARCHAR2,
  PI_OUTPUTTITLE IN VARCHAR2,
  PI_OUTPUTDESC IN VARCHAR2);
```

**Parameters**  This API has standard parameters (see “Standard Parameters” on page 5) and the following parameters:

- **PARAM P_API_VERSION.** The API uses this parameter to compare the version numbers of incoming calls to its current version number, and returns an unexpected error if the versions are incompatible.

- **PARAM P_INIT_MSG_LIST.** Use this parameter to request that the API initialize the message list. By default, this parameter is set to FND_APLG.FALSE, meaning that APIs do not initialize the message list unless specifically instructed by the user.

- **PARAM P_COMMIT.** Use this parameter to instruct the API to commit after performing its function. By default, this parameter is set to FND_APLG.FALSE, meaning that APIs do not commit unless instructed by the user.

- **PARAM P_VALIDATION_LEVEL.** This parameter helps the API to determine those validation steps to execute and those to skip. The curent behavior is that the API performs a full validation.

- **PARAM PI_EXECUTIONSETUP.** The base object type of Execution setup for which a Submit is requested.

- **PARAM PIO_CHANGEDSYSTEMPARAMETERCOLL.** The collection of SYSTEM parameters.
- PARAM PI_TICOLL. The collection of Table Instances referenced from the Object Instance hierarchy with the Currency information as specified by the user.

- PARAM PI_EXEPARAMCOLL. The collection of CDR_PARAMETER_OBJ_TYPE parameters. This can be passed as null, but if you want to overwrite a parameter's value for submission, then populate this with the Parameter object details and assign the DEFAULT_VALUE attribute to the overwritten parameter value.

The other required attributes are COMPANY_ID, OBJ_ID, OBJ_VER and DEFAULT_VALE.

- PARAM PI_EXEOBJCOLL. This is a collection of CDR_PARAMETER_OBJ_TYPE parameters that contains the PRREF_IDs of Programs to be included in the Execution Setup.

For Report Sets and Report Set Entries, enter the PRREF_IDs of those selected Report Sets that may be included for execution. Use PRREF_ID, PRREF_VER of Objects to OBJ_ID, OBJ_VER of CDR_PARAMETER_OBJ_TYPE. Set DEFAULT_VALE as 'Y'. You do not need to assign other attributes except COMPANY_ID, OBJ_ID, OBJ_VER and DEFAULT_VALE.

- PARAM PI_NODEPRREFID. To submit only a single node(RSE), pass its PrrefId; else, pass this parameter as null.

- PARAM PI_NODEPRREFVER. To submit only a single node, pass its PrrefVer; else, pass as null.

- PARAM PI_JOBID. To resubmit a job, pass the previous JobId; else, pass as null.

- PARAM PO_JOBID. The Job ID that is created during the submission.

- PARAM PI_OUTPUTTITLE. The suffix of the output's title.

- PARAM PI_OUTPUTDESC. The suffix of the output's description.

### 20.1.12 Upgrade an Execution Setup

Use this API to upgrade one single Execution Setup.

**Name** CDR_PUB_DF_EXECUTIONSETUP.UpgradeExecSetup

**Signature**

```sql
PROCEDURE UPGRADEEXECSETUP( 
  P_API_VERSION  IN    NUMBER,
  P_INIT_MSG_LIST  IN    VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_COMMIT  IN    VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_VALIDATION_LEVEL  IN    NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
  X_RETURN_STATUS  OUT    VARCHAR2,
  X_MSG_COUNT  OUT    NUMBER,
  X_MSG_DATA  OUT    VARCHAR2,
  PI_ESCOMPID  IN    NUMBER,
  PI_ESOBJID  IN    NUMBER,
  PI_ESOBJVER  IN    NUMBER,
  PO_UPGRADESTATUS  IN OUT    VARCHAR2
);```

**Parameters** This API has standard parameters (see “Standard Parameters” on page 5) and the following parameters:

- **PI_ESCOMPID** (Mandatory) Enter the COMPANY_ID of the Execution Setup.

- **PI_ESOBJID** (Mandatory) Enter the OBJ_ID of the Execution Setup.
Create and Modify Execution Setups

- **PI_ESOBJVER** (Mandatory) Enter the OBJ_VER of the Execution Setup.
- **PO_UPGRADESTATUS** This is an output parameter that indicates whether or not the Execution Setup’s upgrade was successful.

### 20.1.13 Upgrade All Execution Setups

Use this API to upgrade all the Execution Setups associated with an object.

**Name**  CDR_PUB_DF_EXECUTIONSETUP.UpgradeExecSetups

**Signature**

```
PROCEDURE UPGRADEEXECSETUPS(
  P_API_VERSION IN NUMBER,
  P_INIT_MSG_LIST IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_COMMIT IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_VALIDATION_LEVEL IN NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
  X_RETURN_STATUS OUT VARCHAR2,
  X_MSG_COUNT OUT NUMBER,
  X_MSG_DATA OUT VARCHAR2,
  PI_EOICOMPID IN NUMBER,
  PI_EOIOBJID IN NUMBER,
  PI_EOIOBJVER IN NUMBER,
  PO_NOTRUNNABLEESLIST OUT CDR_BASE_OBJ_COLL);
```

**Parameters** This API has standard parameters (see “Standard Parameters” on page 5) and the following parameters:

- **PI_EOICOMPID** (Mandatory) Enter the COMPANY_ID of the executable object instance for which the Execution Setup is intended.
- **PI_EOIOBJID** (Mandatory) Enter the OBJ_ID of the executable object instance for which the Execution Setup is intended.
- **PI_EOIOBJVER** (Mandatory) Enter the OBJ_VER of the executable object instance for which the Execution Setup is intended.
- **PO_NOTRUNNABLEESLIST** This is an output parameter that returns a collection of CDR_BASE_OBJ_TYPE containing Execution Setups whose status has changed from Runnable to Not-Runnable.

### 20.1.14 Make an Execution Setup Active

Use this API to set the status of an Execution Setup to Active.

**Name**  CDR_PUB_DF_EXECUTIONSETUP.SetActive

**Signature**

```
PROCEDURE SETACTIVE(
  P_API_VERSION IN NUMBER,
  P_INIT_MSG_LIST IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_COMMIT IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_VALIDATION_LEVEL IN NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
  X_RETURN_STATUS OUT VARCHAR2,
  X_MSG_COUNT OUT NUMBER,
  X_MSG_DATA OUT VARCHAR2,
  PI_ESCOMPID IN NUMBER,
  PI_ESOBJID IN NUMBER,
  PI_ESOBJVER IN NUMBER,
  PO_NOTRUNNABLEESLIST OUT CDR_BASE_OBJ_COLL);
```

**Parameters** This API has standard parameters (see “Standard Parameters” on page 5) and the following parameters:

- **PI_ESCOMPID** (Mandatory) Enter the COMPANY_ID of the object the Execution Setup is associated with.
- **PI_ESOBJID** (Mandatory) Enter the OBJ_ID of the object the Execution Setup is associated with.
- **PI_ESOBJVER** (Mandatory) Enter the OBJ_VER of the object the Execution Setup is associated with.
Create and Modify Execution Setups

### Parameters
This API has standard parameters (see "Standard Parameters" on page 5) and the following parameters:

- **PI_ESCOMPID** (Mandatory) Enter the COMPANY_ID of the Execution Setup.
- **PI_ESOBJID** (Mandatory) Enter the OBJ_ID of the Execution Setup.
- **PI_ESOBJVER** (Mandatory) Enter the OBJ_VER of the Execution Setup.

#### 20.1.15 Remove an Execution Setup

Use this API to delete an Execution Setup.

**Name**  
CDR_PUB_DF_EXECUTIONSETUP.RemoveExecutionSetup

**Signature**

```sql
PROCEDURE REMOVEEXECUTIONSETUP(
    P_API_VERSION IN NUMBER,
    P_INIT_MSG_LIST IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_COMMIT IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_VALIDATION_LEVEL IN NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
    X_RETURN_STATUS OUT VARCHAR2,
    X_MSG_COUNT OUT NUMBER,
    X_MSG_DATA OUT VARCHAR2,
    PI_EXECUTIONSETUP IN OUT CDR_BASE_OBJ_COLL
);
```

**Parameters**  
This API has standard parameters (see "Standard Parameters" on page 5) and the following parameter:

**PI_EXECUTIONSETUP** (Mandatory) This is a collection of CDR_BASE_OBJ_TYPEs. For each Execution Setup that you want to remove, initialize a CDR_BASE_OBJ_TYPE and then extend the collection.

The following attributes are required: COMPANY_ID,OBJ_ID,OBJ_VER,OBJECT_VERSION_NUMBER,NAMESPACE_OBJ_ID,NAMESPACE_OBJ_VER.
This is a public interface for mapping-related operations; including—creating and modifying mappings at the Table Descriptor and Column levels, creating a Table Descriptor from a Table instance, or a Table instance from a Table Descriptor, and mapping the two. It also includes functions for getting the unique PRREF_ID and PRREF_VER for the executables and Business Areas that contain Table Descriptors. You need these identifiers to run most of the mapping APIs.

21.1 Create and Modify Mappings

This section contains the following topics:

- Section 21.1.1, "Map a Column"
- Section 21.1.2, "Map a Table Descriptor to a Table Instance"
- Section 21.1.3, "Get a Table Instance ID"
- Section 21.1.4, "Create a Table Descriptor from a Table Instance"
- Section 21.1.5, "Create a Table Instance from a Table Descriptor"
- Section 21.1.6, "Modify a Mapping Column"
- Section 21.1.7, "Modify a Mapping at the Table Descriptor Level"
- Section 21.1.8, "Get a PRREF_ID for an Executable in a Workflow"
- Section 21.1.9, "Get a PRREF_ID for an Object in a Work Area"
- Section 21.1.10, "Get a PRREF_ID for a Program in a Report Set"

21.1.1 Map a Column

Use this API to map a Table Descriptor's Columns to a Table instance's Columns using default criteria such as Name, Data Type, and Length. Before you run this API you must map the Table Descriptor and instance and note the Mapping ID and version.

Name  CDR_PUB_DF_MAPPING.GenerateDefaultMapping

Signature

PROCEDURE GENERATEDEFAULTMAPPING(
  P_API_VERSION IN NUMBER,
  P_INIT_MSG_LIST IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_COMMIT IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_VALIDATION_LEVEL IN NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
  X_RETURN_STATUS OUT VARCHAR2,
Create and Modify Mappings

Oracle Life Sciences Data Hub Application Programming Interface Guide

X_MSG_COUNT OUT NUMBER,
X_MSG_DATA OUT VARCHAR2,
PI_MAPPING_NAMING IN CDR_NAMING_VERSION_OBJ_TYPE
);

Parameters
This API has standard parameters (see "Standard Parameters" on page 5) and the following parameter:

PI_MAPPING_NAMING (Mandatory) This is a parameter of table type CDR_NAMING_VERSION_OBJ_TYPE that contains object attributes. Enter values to identify the Mapping that you want to extend to the Column level.

The following attributes are required: OBJ_ID and OBJ_VER.

21.1.2 Map a Table Descriptor to a Table Instance

Use this API to map a Table Descriptor to a Table instance. You specify the Table Descriptor and Table instance and the API maps the columns if it is possible.

Name
CDR_PUB_DF_MAPPING.AutoMapTableDesc

Signature
PROCEDURE AUTOMAPTABLEDESC(
P_API_VERSION IN NUMBER,
P_INIT_MSG_LIST IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
P_COMMIT IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
P_VALIDATION_LEVEL IN NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
X_RETURN_STATUS OUT VARCHAR2,
X_MSG_COUNT OUT NUMBER,
X_MSG_DATA OUT VARCHAR2,
PI_PRREFID IN CDR_PROGRAM_REFS.PRREF_ID%TYPE,
PI_PRREFVER IN CDR_PROGRAM_REFS.PRREF_VER%TYPE,
PI_TDOBJ IN CDR_NAMING_VERSION_OBJ_TYPE,
PI_TIOBJ IN CDR_NAMING_VERSION_OBJ_TYPE
);

Parameters
This API has standard parameters (see "Standard Parameters" on page 5) and the following parameters:

■ PI_PRREFID (Mandatory) Enter the PRREF_ID of the executable object—Program, Load Set, Data Mart, Business Area that owns the Table Descriptor that you want to map. (Use other APIs in this package to get the PRREF_ID.)

■ PI_PRREFVER (Mandatory) Enter the PRREF_VER of the executable object or Business Area that owns the Table Descriptors that you want to map. (Use other APIs in this package to get the PRREF_VER.)

■ PI_TDOBJ (Mandatory) This is a parameter of table type CDR_NAMING_VERSIONS_OBJ_TYPE that contains CDR Naming Version attributes. Enter values to identify the Table Descriptor you want to map.

The required attributes are: COMPANY_ID, OBJ_ID, OBJ_VER.

■ PI_TIOBJ (Mandatory) This is a parameter of table type CDR_NAMING_VERSIONS_OBJ_TYPE that contains CDR Naming Version attributes. Enter values to identify the Table instance to which you want to map the Table Descriptor.

The required attributes are: COMPANY_ID, OBJ_ID, OBJ_VER.
21.1.3 Get a Table Instance ID

Use this API to get the Object ID of the table instance that is mapped to a particular table descriptor. You can invoke this function within a PLSQL program.

While transforming source data into target data in Oracle Health Sciences Data Management Workbench (DMW), it is needed at times, to flag target data. The DMW API used to assign flags to target data requires the Obj ID of the Table Instance as input. This API performs the task of getting the required Obj ID of the target Table Instance, in order that the Flag-related DMW APIs can be invoked.

**Name**  CDR_PUB_DF_MAPPING.Get_Tab_Inst_ID

**Signature**

FUNCTION GETTABINSTID(
PI_TABDESCNAME IN VARCHAR2,
) RETURN NUMBER;

**Return**  Type NUMBER

**Parameters**  This API has the following parameter:

PI_TABDESCNAME (Mandatory). Enter the Oracle Name of the Table Descriptor whose mapped Table Instance ID you want to retrieve.

21.1.4 Create a Table Descriptor from a Table Instance

Use this API to create a Table Descriptor from an existing Table instance and map the two.

**Name**  CDR_PUB_DF_MAPPING.CreateTabDescFromTabInst

**Signature**

PROCEDURE CREATETABDESCFROMTABINST(
P_API_VERSION IN NUMBER,
P_INIT_MSG_LIST IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
P_COMMIT IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
P_VALIDATION_LEVEL IN NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
X_RETURN_STATUS OUT VARCHAR2,
X_MSG_COUNT OUT NUMBER,
X_MSG_DATA OUT VARCHAR2,
PI_PRREFID IN CDR_PROGRAM_REFS.PRREF_ID%TYPE,
PI_PRREFVER IN CDR_PROGRAM_REFS.PRREF_VER%TYPE,
PI_TIOBJ IN CDR_NAMING_VERSION_OBJ_TYPE
);

**Parameters**  This API has standard parameters (see “Standard Parameters” on page 5) and the following parameters:

- **PI_PRREFID**  (Mandatory) Enter the PRREF_ID of the executable object—Program, Load Set, Data Mart or Business Area, in which you want to create a Table Descriptor. (Use other APIs in this package to get this value.)

- **PI_PRREFVER**  (Mandatory) Enter the PRREF_VER of the executable object or Business Area in which you want to create a Table Descriptor. (Use other APIs in this package to get this value.)

- **PI_TIOBJ**  (Mandatory) This is a parameter of table type CDR_NAMING_VERSIONS_OBJ_TYPE that contains CDR Naming Version attributes.
Create and Modify Mappings

Enter values for the Table instance from which you want to create a Table Descriptor. The required attributes are: COMPANY_ID, OBJ_ID, OBJ_VER.

21.1.5 Create a Table Instance from a Table Descriptor

Use this API to create a Table instance from a Table Descriptor and map the two.

**Name**  CDR_PUB_DF_MAPPING.CreateTabInstFromTabDesc

**Signature**

PROCEDURE CREATETABINSTFROMTABDESC(
  P_API_VERSION IN NUMBER,
  P_INIT_MSG_LIST IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_COMMIT IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_VALIDATION_LEVEL IN NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
  X_RETURN_STATUS OUT VARCHAR2,
  X_MSG_COUNT OUT NUMBER,
  X_MSG_DATA OUT VARCHAR2,
  PI_PRREFID IN CDR_PROGRAM_REFS.PRREF_ID%TYPE,
  PI_PRREFVER IN CDR_PROGRAM_REFS.PRREF_VER%TYPE,
  PI_TDOBJ IN CDR_NAMING_VERSION_OBJ_TYPE,
  PI_INSTANCESUBTYPEID IN CDR_NAMINGS.OBJECT_SUBTYPE_ID%TYPE
);

**Parameters**  This API has standard parameters (see "Standard Parameters" on page 5) and the following parameters:

- **PI_PRREFID** (Mandatory) Enter the PRREF_ID of the executable object instance—Program, Load Set, Data Mart, OR Business Area instance that owns the Table Descriptors that you want to map. (Use other APIs in this package to get this value.)

- **PI_PRREFVER** (Mandatory) Enter the PRREF_VER of the executable object instance or Business Area instance that owns the Table Descriptors that you want to map. (Use other APIs in this package to get this value.)

- **PI_TDOBJ** (Mandatory) This is a parameter of table type CDR_NAMING_VERSIONS_OBJ_TYPE that contains CDR Naming Version attributes. Enter values for the Table Descriptor from which you want to create a Table Descriptor. The required attributes are: COMPANY_ID, OBJ_ID, OBJ_VER.

- **PI_INSTANCESUBTYPEID** (Mandatory) Enter a value for the Table instance’s subtype.

21.1.6 Modify a Mapping Column

Use this API to update a mapping at the Column level. You can map currently unmapped Columns, map Table Descriptor Columns to different Columns in the Table instance, change the default value for any Column, and supply or change a format string for a Column. This API enforces all Column mapping rules.

**Name**  CDR_PUB_DF_MAPPING.UpdateMappingColumns

**Signature**

PROCEDURE UPDATEMAPPINGCOLUMNS(
  P_API_VERSION IN NUMBER,
  P_INIT_MSG_LIST IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
21.1.7 Modify a Mapping at the Table Descriptor Level

Use this API to update a mapping at the Table Descriptor level in accordance with all validation rules.

**Name**  
CDR_PUB_DF_MAPPING.ModifyMapping

**Signature**

PROCEDURE UPDATEMAPPINGCOLUMNS(
  P_API_VERSION IN NUMBER,
  P_INIT_MSG_LIST IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_COMMIT IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_VALIDATION_LEVEL IN NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
  X_RETURN_STATUS OUT VARCHAR2,
  X_MSG_COUNT OUT NUMBER,
  X_MSG_DATA OUT VARCHAR2,
  PI_MAPPING_COLUMNS IN CDR_MAPPING_COLUMNS_COLL
);

**Parameters**  
This API has standard parameters (see "Standard Parameters" on page 5) and the following parameter:

**PI_MAPPING_COLUMNS** (Mandatory) This is a collection of CDR_MAPPING_COLUMNS_TYPEs. For each Column that you want to modify, initialize a CDR_MAPPING_COLUMNS_TYPE and then extend the collection. All the attributes are required.

21.1.8 Get a PRREF_ID for an Executable in a Workflow

Use this API to get the PRREF_ID and PRREF_VER for an executable Object instance contained in a Workflow. You need these values to run Mapping APIs.

**Name**  
CDR_PUB_DF_MAPPING.GetPRREFIDforObjUnderWF

**Signature**

PROCEDURE GETPRREFIDFOROBJUNDERWF(
  P_API_VERSION IN NUMBER,
  P_INIT_MSG_LIST IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_COMMIT IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_VALIDATION_LEVEL IN NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
  X_RETURN_STATUS OUT VARCHAR2,
Create and Modify Mappings

21-6
Oracle Life Sciences Data Hub Application Programming Interface Guide

X_MSG_COUNT OUT NUMBER,
X_MSG_DATA OUT VARCHAR2,
PI_SIMPLEOBJ IN CDR_BASE_OBJ_TYPE,
PI_RSIOBJ IN CDR_BASE_OBJ_TYPE,
PI_WFIOBJ IN CDR_BASE_OBJ_TYPE,
PO_PRREFID OUT CDR_PROGRAM_REFS.PRREF_ID%TYPE,
PO_PRREFVER OUT CDR_PROGRAM_REFS.PRREF_VER%TYPE
);

Parameters This API has standard parameters (see "Standard Parameters" on page 5) and the following parameters:

- **PI_SIMPLEOBJ** (Mandatory) This is a parameter of table type CDR_BASE_OBJ_TYPE that contains object attributes. Enter values to identify the Program, Load Set, or Data Mart instance whose PRREFID you need.

  The following attributes are required: COMPANY_ID, OBJ_ID, OBJ_VER.

- **PI_RSIOBJ** (Optional) If you are getting the PRREFID for a Program instance contained in a Report Set that within a Workflow, enter values to identify the Report Set instance in which the Program instance is located. This is a parameter of table type CDR_BASE_OBJ_TYPE that contains object attributes.

  The following attributes are required: COMPANY_ID, OBJ_ID, OBJ_VER.

- **PI_WFIOBJ** (Mandatory) This is a parameter of table type CDR_BASE_OBJ_TYPE that contains object attributes. Enter values to identify the Workflow instance that contains the Program, Load Set, or Data Mart instance whose PRREFID you need.

  The following attributes are required: COMPANY_ID, OBJ_ID, OBJ_VER.

- **PO_PRREFID** This output parameter returns the executable object instance’s PRREF_ID.

- **PO_PRREFVER** This output parameter returns the executable object instance’s PRREF_VER.

21.1.9 Get a PRREF_ID for an Object in a Work Area

Use this API to get the PRREF_ID and PRREF_VER for a Program, Load Set, Data Mart, or Business Area instance contained directly in a Work Area. You need these values to run Mapping APIs.

Name CDR_PUB_DF_MAPPING.GetPRREFIDforSimpleObject

Signature

PROCEDURE GETPRREFIDFORSIMPLEOBJECT(
  P_API_VERSION IN NUMBER,
  P_INIT_MSG_LIST IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_COMMIT IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_VALIDATION_LEVEL IN NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
  X_RETURN_STATUS OUT VARCHAR2,
  X_MSG_COUNT OUT NUMBER,
  X_MSG_DATA OUT VARCHAR2,
  PI_SIMPLEOBJ IN CDR_BASE_OBJ_TYPE,
  PO_PRREFID OUT CDR_PROGRAM_REFS.PRREF_ID%TYPE,
  PO_PRREFVER OUT CDR_PROGRAM_REFS.PRREF_VER%TYPE
);
**Parameters**  This API has standard parameters (see "Standard Parameters" on page 5) and the following parameters:

- **PI_SIMPLEOBJ** (Mandatory) This is a parameter of table type CDR_BASE_OBJ_TYPE that contains object attributes. Enter values to identify the Program, Load Set, Data Mart, or Business Area instance whose PRREFID you need in order to map its Table Descriptors.
  
  The following attributes are required: COMPANY_ID,OBJ_ID,OBJ_VER.

- **PO_PRREFID** This output parameter returns the Program, Load Set, Data Mart, or Business Area instance’s PRREF_ID.

- **PO_PRREFVER** This output parameter returns the Program, Load Set, Data Mart, or Business Area instance’s PRREF_VER.

### 21.1.10 Get a PRREF_ID for a Program in a Report Set

Use this API to get the PRREF_ID and PRREF_VER for a Program instance contained in a Report Set. You need these values to run Mapping APIs.

**Name**  CDR_PUB_DF_MAPPINGS.GetPRREFIDforPgmUnderRSE

**Signature**

```
PROCEDURE GETPRREFIDFORPGMUNDERRSE(
  P_API_VERSION  IN    NUMBER,
  P_INIT_MSG_LIST  IN    VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_COMMIT  IN    VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_VALIDATION_LEVEL  IN    NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
  X_RETURN_STATUS  OUT    VARCHAR2,
  X_MSG_COUNT  OUT    NUMBER,
  X_MSG_DATA  OUT    VARCHAR2,
  PI_SIMPLEOBJ  IN    CDR_BASE_OBJ_TYPE,
  PI_RSIOBJ  IN    CDR_BASE_OBJ_TYPE,
  PO_PRREFID  OUT    CDR_PROGRAM_REFS.PRREF_ID%TYPE,
  PO_PRREFVER  OUT    CDR_PROGRAM_REFS.PRREF_VER%TYPE
);
```

**Parameters**  This API has standard parameters (see "Standard Parameters" on page 5) and the following parameters:

- **PI_SIMPLEOBJ** (Mandatory) This is a parameter of table type CDR_BASE_OBJ_TYPE that contains object attributes. Enter values to identify the Program instance whose PRREFID you need in order to map its Table Descriptors.
  
  The following attributes are required: COMPANY_ID,OBJ_ID,OBJ_VER.

- **PI_RSIOBJ** (Mandatory) This is a parameter of table type CDR_BASE_OBJ_TYPE that contains object attributes. Enter values to identify the Report Set instance that contains the Program instance whose PRREFID you need.
  
  The following attributes are required: COMPANY_ID, OBJ_ID AND OBJ_VER.

- **PO_PRREFID** Output Prref_Id

- **PO_PRREFVER** Output Prref Ver
This package includes APIs for submitting a job for printing, and for retrieving the BLOB and CLOB objects associated with Oracle LSH output.

22.1 Generate Outputs

This section contains the following topics:

- Section 22.1.1, "Submit a Print Request"
- Section 22.1.2, "Get an Output’s BLOB"
- Section 22.1.3, "Get an Output’s CLOB"

22.1.1 Submit a Print Request

Use this API to print a specified output. The API submits a request to the Oracle LSH printing concurrent program that prints the specified output.

**Name**  CDR_PUB_PRINT_OUTPUT.SubmitPrintRequest

**Signature**

FUNCTION SUBMITPRINTREQUEST(
  P_API_VERSION IN NUMBER,
  P_INIT_MSG_LIST IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_COMMIT IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_VALIDATION_LEVEL IN NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
  X_RETURN_STATUS OUT VARCHAR2,
  X_MSG_COUNT OUT NUMBER,
  X_MSG_DATA OUT VARCHAR2,
  PI_NCOMPANYID IN CDR_NAMINGS.COMPANY_ID%TYPE,
  PI_NOUTPUTID IN CDR_NAMINGS.OBJ_ID%TYPE,
  PI_NOUTPUTOBJVER IN CDR_NAMING_VERSIONS.OBJ_VER%TYPE,
  PI_VPRINTERNAME IN FND_PRINTER.PRINTER_NAME%TYPE := NULL,
  PI_VCOVERSHEET IN VARCHAR2 := NULL
) RETURN NUMBER;

**Return**  Type NUMBER

Description Concurrent Request Id

**Parameters**  This API has standard parameters (see "Standard Parameters" on page 5) and the following parameters:

- **PL_NCOMPANYID** (Mandatory) Enter your company ID.
To get your company ID, use CDR_DF_PUB_DEF_CONSTANTS.Current_Company_ID.

- **PI_NOUTPUTID** (Mandatory) Enter the obj_id of the output you want to print.
- **PI_NOUTPUTOBJVER** (Mandatory) Enter the obj_ver of the output you want to print.
- **PI_VPRINTERNAME** (Mandatory) Enter the printer name.
- **PI_VCOVERSHEET** (Mandatory) Enter the text for the coversheet. You can enter text only up to 4000 bytes.

### 22.1.2 Get an Output’s BLOB

Use this API to retrieve the BLOB associated with an output object.

**Name**  CDR_PUB_PRINT_OUTPUT.GetOutPutBlob

**Signature**

FUNCTION GETOUTPUTBLOB(
  P_API_VERSION IN NUMBER,
  P_INIT_MSG_LIST IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_COMMIT IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_VALIDATION_LEVEL IN NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
  PI_NCOMPANYID IN CDR_NAMINGS.COMPANY_ID%TYPE,
  PI_NOUTPUTID IN CDR_NAMINGS.OBJ_ID%TYPE,
  PI_NOUTPUTOBJVER IN CDR_NAMING_VERSIONS.OBJ_VER%TYPE
) RETURN BLOB;

**Return**  Type BLOB

**Description** BLOB

**Parameters**  This API has standard parameters (see "Standard Parameters" on page 5) and the following parameters:

- **PI_NCOMPANYID** (Mandatory) Enter your company ID.
  
  To get your company ID, use CDR_DF_PUB_DEF_CONSTANTS.Current_Company_ID.

- **PI_NOUTPUTID** (Mandatory) Enter the obj_ID of the output whose BLOB you want to retrieve.

- **PI_NOUTPUTOBJVER** (Mandatory) Enter the obj_ver of the output whose BLOB you want to retrieve.

### 22.1.3 Get an Output’s CLOB

Use this API to retrieve the CLOB associated with a specified output object.

**Name**  CDR_PUB_PRINT_OUTPUT.GetOutputCLOB

**Signature**

FUNCTION GETOUTPUTCLOB(
  P_API_VERSION IN NUMBER,
  P_INIT_MSG_LIST IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_COMMIT IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_VALIDATION_LEVEL IN NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
PI_NCOMPANYID IN CDR_NAMINGS.COMPANY_ID%TYPE,
PI_NOUTPUTID IN CDR_NAMINGS.OBJ_ID%TYPE,
PI_NOUTPUTOBJVER IN CDR_NAMING_VERSIONS.OBJ_VER%TYPE
RETURN CLOB;

Return  Type CLOB

Description Clob

Parameters  This API has standard parameters (see "Standard Parameters" on page 5) and the following parameters:

- **PI_NCOMPANYID** (Mandatory) Enter your company ID. To get your company ID, use CDR_DF_PUB_DEF_CONSTANTS.Current_Company_ID.
- **PI_NOUTPUTID** (Mandatory) Enter the obj_ID of the output whose CLOB you want to retrieve.
- **PI_NOUTPUTOBJVER** (Mandatory) Enter the obj_ver of the output whose CLOB you want to retrieve.
This is a public interface which hosts the Naming API for updating the Version label.

### 23.1 Modify Version Labels

This section contains one API for updating a version label.

### 23.1.1 Update a Version Label

Use this API to create or modify an object version label. If the same label exists for another version of the object, the API returns that version number. * @param p_api_version (Mandatory) Enter the current version of the API you are calling. The API compares the version numbers of incoming calls to its current version number and returns an error if they are incompatible.

**Name**  
CDR_PUB_DF_NAMING.UpdateVersionLabel

**Signature**

```
PROCEDURE UPDATEVERSIONLABEL(
  P_API_VERSION  IN    NUMBER,
  P_INIT_MSG_LIST  IN    VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_COMMIT  IN    VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_VALIDATION_LEVEL  IN    NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
  X_RETURN_STATUS  OUT    VARCHAR2,
  X_MSG_COUNT  OUT    NUMBER,
  X_MSG_DATA  OUT    VARCHAR2,
  PIO_NAMING  IN OUT    CDR_NAMING_VERSION_OBJ_TYPE
);
```

**Parameters**  
This API has standard parameters (see "Standard Parameters" on page 5) and the following parameters:

**PIO_NAMING** (Mandatory) This is a parameter of table type CDR_NAMING VERSION_OBJ_TYPE that contains object attributes. Enter values to identify the object version that you want to label.

The following attributes are required: COMPANY_ID,OBJ_ID,OBJ_VER,VERSION_LABEL. For VERSION_LABEL enter the text of the label you want to apply to this object version.
This section contains the following topics:

- Section 24.1, "Classify Objects"
- Section 24.2, "Classify Subtypes"
- Section 24.3, "Create and Modify Classification Hierarchy Values"

24.1 Classify Objects

This is a public interface for assigning and removing object classifications.

This section contains the following topics:

- Section 24.1.1, "Classify an Object"
- Section 24.1.2, "Declassify an Object"

24.1.1 Classify an Object

Use this API to classify Objects

**Name**  
CDR_PUB_CLA_OBJ_CLASSIFICATION.AssignObjectClassification

**Signature**

PROCEDURE ASSIGNOBJECTCLASSIFICATION(
  P_API_VERSION  IN    NUMBER,
  P_INIT_MSG_LIST  IN    VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_COMMIT  IN    VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_VALIDATION_LEVEL  IN    NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
  X_RETURN_STATUS  OUT    VARCHAR2,
  X_MSG_COUNT  OUT    NUMBER,
  X_MSG_DATA  OUT    VARCHAR2,
  PIO_SOURCECDRNAMING  IN OUT    CDR_NAMING_VERSION_OBJ_TYPE,
  PI_OCLASSIFICATIONS  IN    CDR_CLASSIFICATIONS_COLL
);

**Parameters**  
This API has standard parameters (see "Standard Parameters" on page 5) and the following parameters:

- **PIO_SOURCECDRNAMING** (Mandatory) This is a parameter of table type CDR_NAMING_VERSION_OBJ_TYPE that contains object attributes. Enter values to identify the object that you want to classify.
The following attributes are required: COMPANY_ID, OBJ_ID, OBJ_VER, OBJECT_ VERSION_NUMBER, NAMESPACE_OBJ_ID, NAMESPACE_OBJ_VER, OBJECT_ SUBTYPE_ID.

- **PI_OCLASSIFICATIONS** This is a collection of CDR_CLA_OBJ_TYPES, which have 5 attributes, including CLA_LEVEL_ID and CLASSIFICATION_ID.

If you want the OBJECT to inherit its classifications for a particular level from its parent, enter the classification level ID and, for the CLASSIFICATION_ID, enter 0 (zero).

If you want to explicitly assign one or more terms for a particular level, initialize a CDR_CLA_OBJ_TYPE for each term, entering the classification level ID and, for the CLASSIFICATION_ID, the term ID. The PAR_attributes are relevant only to Planned Outputs.

If you are not classifying a Planned Output, do not enter any values for them.

If you are classifying a Planned Output and want to use a Parameter with a classification-driven list of values, use the PAR_attributes to identify the Parameter you want to use.

### 24.1.2 Declassify an Object

Use this procedure to remove a single classification value from a defined Object.

**Name** CDR_PUB_CLA_OBJ_CLASSIFICATION.Declassify

**Signature**

```
PROCEDURE DECLASSIFY(
    P_API_VERSION  IN    NUMBER,
    P_INIT_MSG_LIST  IN    VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_COMMIT  IN    VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_VALIDATION_LEVEL  IN    NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
    X_RETURN_STATUS  OUT    VARCHAR2,
    X_MSG_COUNT  OUT    NUMBER,
    X_MSG_DATA  OUT    VARCHAR2,
    PI_NOBJECTID  IN    CDR_NAMINGS.OBJ_ID%TYPE,
    PI_NCLALEVELID  IN    PLS_INTEGER,
    PI_NCLASSIFICATIONID  IN    PLS_INTEGER
);
```

**Parameters** This API has standard parameters (see "Standard Parameters" on page 5) and the following parameters:

- **PI_NOBJECTID** (Mandatory) Enter the object ID of the object from which you want to remove classifications.

- **PI_NCLALEVELID** (Mandatory) Enter the ID of the classification hierarchy level that contains the classification value, or term, that you want to remove.

- **PI_NCLASSIFICATIONID** (Mandatory) Enter the ID of the classification value, or term, that you want to remove.

### 24.2 Classify Subtypes

This is a public interface for all Classification functions related to Subtypes.

This section contains the following topics:
Classify Subtypes

- Section 24.2.1, "Get a Subtype Classification Level"
- Section 24.2.2, "Get an Object Classification Value"
- Section 24.2.3, "Get a Parent Term"

24.2.1 Get a Subtype Classification Level

Use this API to retrieve all the classification hierarchy levels assigned to an Object Subtype. The function returns a collection of CDR_HIER_LEVEL_VAL_OBJ_TYPE.

Name CDR_PUB_CLA_SUBTYPES.GetClassificationLevels

Signature

FUNCTION GETCLASSIFICATIONLEVELS(
  P_API_VERSION IN NUMBER,
  P_INIT_MSG_LIST IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_COMMIT IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_VALIDATION_LEVEL IN NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
  PIOBJECTSUBTYPEID IN CDR_OBJECT_SUBTYPES_TL.OBJECT_SUBTYPE_ID%TYPE
) RETURN CDR_HIER_LEVEL_VALUES_COLL;

Return

Type CDR_HIER_LEVEL_VALUES_COLL
Description classification levels.

Parameters This API has standard parameters (see "Standard Parameters" on page 5) and the following parameter:

PIOBJECTSUBTYPEID Enter the ID of the Object Subtype whose assigned classification levels you want to retrieve.

24.2.2 Get an Object Classification Value

Use this API to get the classifications assigned to the given object ID and version.

Name CDR_PUB_CLA_SUBTYPES.GetObjectClaValues

Signature

FUNCTION GETOBJECTCLAVALUES(
  P_API_VERSION IN NUMBER,
  P_INIT_MSG_LIST IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_COMMIT IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_VALIDATION_LEVEL IN NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
  POBJECTID IN CDR_OBJ_CLA_MAPPINGS.OBJ_ID%TYPE,
  PIOBJECTSUBTYPEID IN CDR_OBJECT_SUBTYPES_TL.OBJECT_SUBTYPE_ID%TYPE
) RETURN CDR_HIER_LEVEL_VALUES_COLL;

Return

Type CDR_HIERLEVEL_VALUES_COLL
Description Classification values

Parameters This API has standard parameters (see "Standard Parameters" on page 5) and the following parameters:

- POBJECTID (Mandatory) Enter the Object Id.
- **PIOBJECTSUBTYPEID** (Mandatory) Enter the Object Subtype Id.

### 24.2.3 Get a Parent Term

Use this API to retrieve some or all of the terms on a particular level that are children of a particular parent term. The function returns a collection of CDR_HIER_LEVEL_VAL_OBJ_TYPES.

**Name**  CDR_PUB_CLA_SUBTYPES.GetClaHierValues

**Signature**

FUNCTION GETCLAHIERVALUES(
  P_API_VERSION  IN    NUMBER,
  P_INIT_MSG_LIST  IN    VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_COMMIT  IN    VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_VALIDATION_LEVEL  IN    NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
  PLEVELIDS  IN    CDR_OPA_ID_COLL,
  PTERMS  IN    CDR_OPA_STRING_COLL,
  PDOMAINID  IN    PLS_INTEGER,
  PCLALEVELID  IN    CDR_SUBTYPE_CLA_LEVELS.CLA_LEVEL_ID%TYPE
) RETURN CDR_HIER_LEVEL_VALUES_COLL;

**Return Type** CDR_HIER_LEVEL_VALUES_COLL

**Description** Classification hierarchy values

**Parameters**  This API has standard parameters (see “Standard Parameters” on page 5) and the following parameters:

- **PLEVELIDS** (Mandatory) This is a collection of CDR_OPA_ID_OBJ_TYPES. For each hierarchy level from the top level to the level from which you want to retrieve terms (but not lower), initialize a CDR_OPA_ID_OBJ_TYPE and then extend the collection.

  The following attribute is required: CDR_OPA_ID.

- **PTERMS** (Mandatory) This is a collection of CDR_OPA_STRING_OBJ_TYPES. For each hierarchy level from the top level to the one from which you want to retrieve terms (but not lower), initialize a CDR_OPA_ID_OBJ_TYPE and then extend the collection. For the required attribute, CDR_OPA_STRING, enter the term whose related terms you want to retrieve, in order starting with the top level.

- **PDOMAINID** Enter the ID of the LSH Instance Domain. Use the following query to get this ID: select * from TMS.TMS_DEF_DOMAINS where name = 'CDR_USER_HIER'

- **PCLALEVELID** Enter the ID of the classification hierarchy level that contains the terms you are searching for.

### 24.3 Create and Modify Classification Hierarchy Values

This is a public interface for classification hierarchy value-related operations.

This section contains the following topics:

- **Section 24.3.1, "Insert a Classification Value"**
- **Section 24.3.2, "Update a Classification Value"**
24.3.1 Insert a Classification Value

Use this API to insert values, or terms, into a classification hierarchy level, related to a term in the next higher level. The function returns 0 if the transaction failed and 1 if it succeeded.

**Name**  CDR_PUB_CLA_HIERARCHY_VALS.InsertValues

**Signature**

FUNCTION INSERTVALUES(
  P_API_VERSION IN NUMBER,
  P_INIT_MSG_LIST IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_COMMIT IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_VALIDATION_LEVEL IN NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
  PPARENTCONTENTID IN TMS.TMS_DICT_CONTENTS.DICT_CONTENT_ID%TYPE,
  PLEVELID IN TMS.TMS_DEF_LEVELS.DEF_LEVEL_ID%TYPE,
  PV VALUES IN OUT CDR_HIER_VAL_COLL
)
RETURN PLS_INTEGER;

**Return**

Type PLS_INTEGER

Description 0 is returned if the transaction fails

1 is returned if the transaction succeeds.

**Parameters**  This API has standard parameters (see "Standard Parameters" on page 5) and the following parameters:

- **PPARENTCONTENTID** Enter the ID of a term that serves as a parent to the terms you are inserting.

- **PLEVELID** Enter the ID of the classification hierarchy level into which you want to insert the terms.

- **PV VALUES** (Mandatory) This is a collection of CDR_HIER_VAL_TYPEs. For each classification value (term) that you want to add as a child to the term you specified, initialize a CDR_HIER_VAL_TYPE and then extend the collection.

  The following attributes are required: CONTENT_ID,TERM,ERROR_MSG,APPROVED_FLAG.

24.3.2 Update a Classification Value

Use this API to modify classification hierarchy values. It returns 1 for transaction success and 0 for failure. It also returns a PV VALUES collection that includes error messages for any terms that could not be updated.

**Name**  CDR_PUB_CLA_HIERARCHY_VALS.UpdateValues

**Signature**

FUNCTION UPDATEVALUES(
  P_API_VERSION IN NUMBER,
  P_INIT_MSG_LIST IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_COMMIT IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_VALIDATION_LEVEL IN NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
Create and Modify Classification Hierarchy Values

```plaintext
PVALUES IN OUT CDR_HIER_VAL_COLL
) RETURN PLS_INTEGER;

Return
Type PLS_INTEGER
Description 0 is returned if the transaction fails (Values will contain the errors).
1 is returned if the transaction succeeds.

Parameters This API has standard parameters (see "Standard Parameters" on page 5) and the following parameter:

PVALUES (Mandatory) This is a collection of CDR_HIER_VAL_TYPEs. For each classification value (term) that you want to modify, initialize a CDR_HIER_VAL_TYPE and then extend the collection.

The following attributes are required: CONTENT_ID,TERM,APPROVED_FLAG.
You can modify the term itself or its Approved flag value.

24.3.3 Delete a Classification Value
Use this API to delete classification values (terms). The API does not delete terms if they have been assigned as the default value for an object subtype or if they have been assigned to any object. If neither case is true, then the API deletes the term and any related terms it may have lower in the classification hierarchy.

Name CDR_PUB_CLA_HIERARCHY_VALS.DeleteValues

Signature
PROCEDURE DELETEVALUES(
    P_API_VERSION IN NUMBER,
P_INIT_MSG_LIST IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
P_COMMIT IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
P_VALIDATION_LEVEL IN NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
X_RETURN_STATUS OUT VARCHAR2,
X_MSG_COUNT OUT NUMBER,
X_MSG_DATA OUT VARCHAR2,
PVALUES IN OUT CDR_HIER_VAL_COLL
);

Parameters This API has standard parameters (see "Standard Parameters" on page 5) and the following parameter:

PVALUES (Mandatory) This is a collection of CDR_HIER_VAL_TYPEs. For each classification value (term) that you want to delete, initialize a CDR_HIER_VAL_TYPE and then extend the collection.

The following attribute is required: CONTENT_ID.
This section contains the following topics:

- Section 25.1, "Create and Execute Output Jobs"
- Section 25.2, "Retrieve Information about Ongoing Jobs"
- Section 25.3, "Set Execution Statuses"
- Section 25.4, "Submit Messages"
- Section 25.5, "Create Submission Records"

### 25.1 Create and Execute Output Jobs

This is a public interface that hosts the API for uploading output CLOBs into Oracle LSH. These output CLOBs are generated by Oracle LSH executables outside the Oracle LSH database. For example, a PDF file output from a SAS print program; or a SAS CPORT file output from a Data Mart. This public interface also includes APIs that adapters may need to call during the execution of a Load Set, Data Mart, or Program.

This section contains the following topics:

- Section 25.1.1, "Create a Binary Output"
- Section 25.1.2, "Upload an Output BLOB"
- Section 25.1.3, "Upload an Output Clob"
- Section 25.1.4, "Upload a LOB to a Temporary Table"
- Section 25.1.5, "Download a Job Output BLOB"
- Section 25.1.6, "Queue a Job"
- Section 25.1.7, "Wait for a Job to Complete"
- Section 25.1.8, "Generate an XML Payload"

### 25.1.1 Create a Binary Output

Use this API to create a binary output object.

**Name**  
CDR_PUB_EXE_EXTERNAL.CreateBinaryOutput

**Signature**

```sql
PROCEDURE CREATEBINARYOUTPUT(
    P_API_VERSION IN    NUMBER,
    P_INIT_MSG_LIST IN    VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
```

```sql
```
Create and Execute Output Jobs

```sql
P_COMMIT IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
P_VALIDATION_LEVEL IN NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
X_RETURN_STATUS OUT VARCHAR2,
X_MSG_COUNT OUT NUMBER,
X_MSG_DATA OUT VARCHAR2,
PI_NCOMPANYID IN NUMBER,
PI_VFILENAME IN VARCHAR2,
PI_VDATA IN RAW := NULL
);
```

**Parameters**  This API has standard parameters (see "Standard Parameters" on page 5) and the following parameters:

- **PI_NCOMPANYID** (Mandatory) Enter Company Id.
- **PI_VFILENAME** (Mandatory) Enter the File Name.
- **PI_VDATA** (Optional) Enter the RAW input to stream up the output.

### 25.1.2 Upload an Output BLOB

Use this API to upload an output BLOB generated by an external processing engine into Oracle LSH.

**Name**  CDR_PUB_EXEEXTERNAL.UploadBlobOutput

**Signature**

```sql
PROCEDURE UPLOADBLOBOUTPUT(
  P_API_VERSION IN NUMBER,
  P_INIT_MSG_LIST IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_COMMIT IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_VALIDATION_LEVEL IN NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
  X_RETURN_STATUS OUT VARCHAR2,
  X_MSG_COUNT OUT NUMBER,
  X_MSG_DATA OUT VARCHAR2,
  PI_NJOBID IN VARCHAR2,
  PI_VFILENAME IN VARCHAR2,
  PI_NPRREFID IN VARCHAR2,
  PIO_BLOBSTREAM IN OUT BLOB
);
```

**Parameters**  This API has standard parameters (see "Standard Parameters" on page 5) and the following parameters:

- **PI_NJOBID** (Mandatory) Enter the JOB_ID of the job that generated the output.
- **PI_VFILENAME** (Mandatory) Enter the filename associated with the Planned Output.
- **PI_NPRREFID** (Mandatory) Enter the PRREF_ID of the Oracle LSH executable. This attribute is available from the cdr_jobs_v view.
- **PIO_BLOBSTREAM** (Mandatory) This is a parameter of type BLOB. Enter BLOB to be uploaded. @rep:scope public.

### 25.1.3 Upload an Output Clob

Use this API to upload an output CLOB into Oracle LSH.

**Name**  CDR_PUB_EXE_EXTERNAL.UploadClobOutput

```sql
P_API_VERSION IN NUMBER,
P_INIT_MSG_LIST IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
P_COMMIT IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
P_VALIDATION_LEVEL IN NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
X_RETURN_STATUS OUT VARCHAR2,
X_MSG_COUNT OUT NUMBER,
X_MSG_DATA OUT VARCHAR2,
PI_NJOBID IN VARCHAR2,
PI_VFILENAME IN VARCHAR2,
PI_NPRREFID IN VARCHAR2,
PIO_BLOBSTREAM IN OUT BLOB
);
```

**Parameters**  This API has standard parameters (see "Standard Parameters" on page 5) and the following parameters:

- **PI_NJOBID** (Mandatory) Enter the JOB_ID of the job that generated the output.
- **PI_VFILENAME** (Mandatory) Enter the filename associated with the Planned Output.
- **PI_NPRREFID** (Mandatory) Enter the PRREF_ID of the Oracle LSH executable. This attribute is available from the cdr_jobs_v view.
- **PIO_BLOBSTREAM** (Mandatory) This is a parameter of type BLOB. Enter BLOB to be uploaded. @rep:scope public.
Signature

PROCEDURE UPLOADCLOBOUTPUT(
  P_API_VERSION IN NUMBER,
  P_INIT_MSG_LIST IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_COMMIT IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_VALIDATION_LEVEL IN NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
  X_RETURN_STATUS OUT VARCHAR2,
  X_MSG_COUNT OUT NUMBER,
  X_MSG_DATA OUT VARCHAR2,
  PI_NJOBID IN VARCHAR2,
  PI_VFILENAME IN VARCHAR2,
  PI_NPRREFID IN VARCHAR2,
  PIO_CLOBSTREAM IN OUT CLOB
);

Parameters
This API has standard parameters (see “Standard Parameters” on page 5) and the following parameters:

- **PI_NJOBID** (Mandatory) Enter the JOB_ID of the job that generated the output.
- **PI_VFILENAME** (Mandatory) Enter the filename associated with the Planned Output.
- **PI_NPRREFID** (Mandatory) Enter the PRREFID of the Oracle LSH executable. This attribute is available from the CDR_JOBS table.
- **PIO_CLOBSTREAM** (Mandatory) Enter the variable name for the CLOB in the database.

25.1.4 Upload a LOB to a Temporary Table

Use this API to upload one or more BLOBs or CLOBs (binary or character large objects) to a temporary table.

Name  CDR_PUB_EXE_EXTERNAL.CreateTempLobs

Signature

PROCEDURE CREATETEMPLOBS(
  P_API_VERSION IN NUMBER,
  P_INIT_MSG_LIST IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_COMMIT IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_VALIDATION_LEVEL IN NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
  X_RETURN_STATUS OUT VARCHAR2,
  X_MSG_COUNT OUT NUMBER,
  X_MSG_DATA OUT VARCHAR2,
  PI_NJOBID IN NUMBER,
  PI_BZIP IN BOOLEAN,
  PI_BLOBNAMES IN CDR_VC_LIST_COLL,
  PI_BLOBENTRIES IN CDR_BLOB_LIST_COLL,
  PI_CLOBNAMES IN CDR_VC_LIST_COLL,
  PI_CLOBENTRIES IN CDR_CLOB_LIST_COLL
);

Parameters
This API has standard parameters (see “Standard Parameters” on page 5) and the following parameters:

- **PI_NJOBID** (Mandatory) Enter the Job ID. You can get the Job ID by running CDR_PUB_EXE_RUNTIME.GETCURRENTLYEXECUTINGJOBID.
- **PI_BZIP** Is the input lobs in zip format? Enter TRUE or FALSE.
Create and Execute Output Jobs

- **PI_BLOBNAMES** This parameter is of type cdr_vc_list_coll which is a collection of varchar2(2000). The varchar contains the BLOB file name.

- **PI_BLOBENTRIES** This parameter is a collection of cdr_blob_list_coll which is a collection of BLOB types. This parameter contains the actual file BLOBs. This is always a collection so even if you are uploading a single CLOB, initialize the collection with that BLOB and pass it here.

- **PI_CLOBNAMES** This parameter is of type cdr_vc_list_coll which is a collection of varchar2(2000). The varchar contains the CLOB file name.

- **PI_CLOBENTRIES** This parameter is a collection of cdr_clob_list_coll which is a collection of CLOB type. This contains the actual file CLOBs. This is always a collection so even if you are uploading a single CLOB, initialize the collection with that CLOB and pass it here.

25.1.5 Download a Job Output BLOB

Use this API to download a job output BLOB.

**Name**  CDR_PUB_EXE_EXTERNAL.DownloadTempBlob

**Signature**

FUNCTION DownloadTempBlob(
  P_API_VERSION IN NUMBER,
  P_INIT_MSG_LIST IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_COMMIT IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_VALIDATION_LEVEL IN NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
  PI_NOBJID IN NUMBER,
  PO_VFILENAME OUT VARCHAR2
) RETURN BLOB;

**Return**  Type BLOB

Description returns the output lob file as a BLOB.

**Parameters**  This API has standard parameters (see "Standard Parameters" on page 5) and the following parameters:

- **PI_NOBJID** (Mandatory) Enter the tmp_blob_id of the job output LOB. You can use the following query to get this ID:

  SELECT tmp_blob_id FROM cdr_temp_blobs_v WHERE job_id = pi_nJobId;

- **PO_VFILENAME** (Mandatory) This Out parameter contains the file name of the downloaded lob file.

25.1.6 Queue a Job

Use this API to queue the job into a service location for execution.

**Name**  CDR_PUB_EXE_EXTERNAL.SendJob

**Signature**

PROCEDURE SendJob(
  P_API_VERSION IN NUMBER,
  P_INIT_MSG_LIST IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_COMMIT IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_VALIDATION_LEVEL IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_VALIDATE_LEVEL_COMPLETE,
  X_RETURN_STATUS OUT VARCHAR2,
  X_RESULT_OUT OUT VARCHAR2,
  X_RESULT_IN OUT VARCHAR2,
  X_FINAL_RESULT OUT VARCHAR2,
  X_FINAL_RESULT_IN OUT VARCHAR2,
  X_FINAL_RESULT_OUT OUT VARCHAR2,
  X_FINAL_RESULT_OUT_IN OUT VARCHAR2,
  X_FINAL_RESULT_OUT_OUT OUT VARCHAR2,
  X_FINAL_RESULT_OUT_OUT_IN OUT VARCHAR2,
  X_FINAL_RESULT_OUT_OUT_OUT OUT VARCHAR2,
  X_FINAL_RESULT_OUT_OUT_OUT_IN OUT VARCHAR2,
  X_FINAL_RESULT_OUT_OUT_OUT_OUT OUT VARCHAR2,
  X_FINAL_RESULT_OUT_OUT_OUT_OUT_IN OUT VARCHAR2,
  X_FINAL_RESULT_OUT_OUT_OUT_OUT_OUT OUT VARCHAR2,
  X_FINAL_RESULT_OUT_OUT_OUT_OUT_OUT_IN OUT VARCHAR2,
  X_FINAL_RESULT_OUT_OUT_OUT_OUT_OUT_OUT OUT VARCHAR2,
  X_FINAL_RESULT_OUT_OUT_OUT_OUT_OUT_OUT_IN OUT VARCHAR2,
  X_FINAL_RESULT_OUT_OUT_OUT_OUT_OUT_OUT_OUT OUT VARCHAR2,
  X_FINAL_RESULT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_IN OUT VARCHAR2,
  X_FINAL_RESULT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT OUT VARCHAR2,
  X_FINAL_RESULT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_IN OUT VARCHAR2,
  X_FINAL_RESULT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT OUT VARCHAR2,
  X_FINAL_RESULT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_IN OUT VARCHAR2,
  X_FINAL_RESULT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT OUT VARCHAR2,
  X_FINAL_RESULT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_IN OUT VARCHAR2,
  X_FINAL_RESULT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT OUT VARCHAR2,
  X_FINAL_RESULT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_IN OUT VARCHAR2,
  X_FINAL_RESULT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT OUT VARCHAR2,
  X_FINAL_RESULT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_IN OUT VARCHAR2,
  X_FINAL_RESULT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT OUT VARCHAR2,
  X_FINAL_RESULT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_IN OUT VARCHAR2,
  X_FINAL_RESULT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT OUT VARCHAR2,
  X_FINAL_RESULT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_IN OUT VARCHAR2,
  X_FINAL_RESULT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT OUT VARCHAR2,
  X_FINAL_RESULT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_IN OUT VARCHAR2,
  X_FINAL_RESULT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT OUT VARCHAR2,
  X_FINAL_RESULT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_IN OUT VARCHAR2,
  X_FINAL_RESULT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT OUT VARCHAR2,
  X_FINAL_RESULT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_IN OUT VARCHAR2,
  X_FINAL_RESULT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT OUT VARCHAR2,
  X_FINAL_RESULT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_IN OUT VARCHAR2,
  X_FINAL_RESULT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT OUT VARCHAR2,
  X_FINAL_RESULT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_IN OUT VARCHAR2,
  X_FINAL_RESULT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT OUT VARCHAR2,
  X_FINAL_RESULT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_IN OUT VARCHAR2,
  X_FINAL_RESULT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT OUT VARCHAR2,
  X_FINAL_RESULT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_IN OUT VARCHAR2,
  X_FINAL_RESULT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT OUT VARCHAR2,
  X_FINAL_RESULT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_IN OUT VARCHAR2,
  X_FINAL_RESULT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT OUT VARCHAR2,
  X_FINAL_RESULT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_OUT_IN OUT VARCHAR2,
Create and Execute Output Jobs

25.1.7 Wait for a Job to Complete

Use this API to enable synchronous execution so that program control waits for a Job to complete before proceeding with the rest of the logic.

**Name**  CDR_PUB_EXE_EXTERNAL.WaitForFinalStatus

**Signature**

```
PROCEDURE WAITFORFINALSTATUS(
    P_API_VERSION  IN    NUMBER,
    P_INIT_MSG_LIST  IN    VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_COMMIT  IN    VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_VALIDATION_LEVEL  IN    NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
    X_RETURN_STATUS  OUT    VARCHAR2,
    X_MSG_COUNT  OUT    NUMBER,
    X_MSG_DATA  OUT    VARCHAR2,
    PI_NJOBID  IN    NUMBER,
    PI_NTIMEOUT  IN    NUMBER,
    PI_NSERVICEINSTANCEID  IN    NUMBER
);
```

**Parameters**  This API has standard parameters (see "Standard Parameters" on page 5) and the following parameters:

- **PI_NJOBID** (Mandatory) Enter the Job ID of the Job in process. The Job_ID is returned by the API CDR_PUB_EXE_SUBMISSION.CREATE_SUBMISSION and in the UI.
- **PI_NTIMEOUT** (Mandatory) Enter in seconds, the job completion period before the API times out.
- **PI_NSERVICEINSTANCEID** (Mandatory) Enter the service instance that is processing the job. Get the ID from the cdr_jobs_v view using the Job_Id.

25.1.8 Generate an XML Payload

Use this API to generate the required XML payload for a job execution.

**Name**  CDR_PUB_EXE_EXTERNAL.GenerateXmlPayload

**Signature**

```
CREATE AND EXECUTE OUTPUT JOBS

FUNCTION GENERATEXMLPAYLOAD(
    P_API_VERSION IN NUMBER,
    P_INIT_MSG_LIST IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_COMMIT IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_VALIDATION_LEVEL IN NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
    PI_NJOBID IN NUMBER,
    PI_VJOBTYPE IN VARCHAR2,
    PI_NDOWNLOADCONFIGID IN NUMBER,
    PI_VPROGRAM IN VARCHAR2,
    PI_VWORKDIRECTORY IN VARCHAR2,
    PI_VOUTPUTPATH IN VARCHAR2,
    PI_VPRIORITY IN VARCHAR2,
    PI_VSCHEMA IN VARCHAR2,
    PI_VUSERID IN VARCHAR2,
    PI_VSUBDIRECTORIES IN CDR_VC_LIST_COLL,
    PI_NSURROGATEJOBID IN CDR_JOBS_V.JOB_ID%TYPE := NULL,
    PI_VTECHTYPE IN VARCHAR2 := NULL,
    PI_NSURROGATEPRREFID IN CDR_JOBS_V.PRREF_ID%TYPE := NULL
) RETURN CLOB;

Return Type CLOB

Description Returns the generated XML as a CLOB.

Parameters This API has standard parameters (see "Standard Parameters" on page 5) and the following parameters:

- **PI_NJOBID** (Mandatory) Enter the Job_ID.
- **PI_VJOBTYPE** (Mandatory) Enter the Job Type—either *tmp* for an internal job or *exe* for a user-executed job.
- **PI_NDOWNLOADCONFIGID** (Optional) If the entire job configuration is stored in cdr_temp_blobs (against the Job_ID), you can enter the ID of the BLOB for the job.

Query cdr_temp_blobs_v to get the value. If you do not enter a BLOB ID, enter 0.
- **PI_VPROGRAM** (Mandatory) Enter the command to be executed on the server.
- **PI_VWORKDIRECTORY** (Mandatory) Enter the full path of the root folder in the DP server Home under which the job folder gets created.
- **PI_VRUNSCRIPT** (Mandatory) Enter the name of the main script to be executed for the job.
- **PI_VOUTPUTPATH** (Mandatory) Enter the path where the output will be generated, relative to the work directory / folder.
- **PI_VPRIORITY** (Mandatory) Enter the job priority. The possible values are the lookup codes in the lookup type 'CDR_JOB_PRIORITIES'.
- **PI_VSCHEMA** (Mandatory) Enter the ID of the ZZ_Schema allocated for the current job. You can query CDR_JOBS_V to get this ID.
- **PI_VUSERID** (Mandatory) Enter the UserId that is executing the job. Use the API CDR_PUB_DEF_FACTORY_SUPPORT.GETUSERID to get this ID.
- **PI_VSUBDIRECTORIES** This is a collection of type CDR_VC_LIST_COLL. Enter the names of subdirectories that should be created under the Job directory in the DP Server.
Retrieve Information about Ongoing Jobs

This is a public interface to retrieve information about an ongoing job.

This section contains the following topics:

- Section 25.2.1, "Get an Ongoing Job ID"
- Section 25.2.2, "Get Currently Executing Parameters"
- Section 25.2.3, "Get Information About a Job"
- Section 25.2.4, "Get Job Information (Overloaded)"

25.2 Retrieve Information about Ongoing Jobs

Use this API to retrieve the job ID of the job that is currently running.

**Name**  CDR_PUB_EXE_RUNTIME.GetCurrentlyExecutingJobID

**Signature**

FUNCTION GETCURRENTLYEXECUTINGJOBID(
    P_API_VERSION  IN    NUMBER,
    P_INIT_MSG_LIST  IN    VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_COMMIT  IN    VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_VALIDATION_LEVEL  IN    NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL
) RETURN CDR_JOBS.JOB_ID%TYPE;

**Return**  Type CDR_JOBS.JOB_ID%TYPE

Description the job ID of the job which is executing in the current session.

**Parameters**  This API has standard parameters. See "Standard Parameters" on page 5) for details.

25.2.1 Get an Ongoing Job ID

Use this API to retrieve the job ID of the job that is currently running.

**Name**  CDR_PUB_EXE_RUNTIME.GetCurrentlyExecutingJobID

**Signature**

FUNCTION GETCURRENTLYEXECUTINGJOBID(
    P_API_VERSION  IN    NUMBER,
    P_INIT_MSG_LIST  IN    VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_COMMIT  IN    VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_VALIDATION_LEVEL  IN    NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL
) RETURN CDR_JOBS.JOB_ID%TYPE;

**Return**  Type CDR_JOBS.JOB_ID%TYPE

Description the job ID of the job which is executing in the current session.

**Parameters**  This API has standard parameters. See "Standard Parameters" on page 5) for details.

25.2.2 Get Currently Executing Parameters

Use this API to retrieve the parameters in use by the job that is currently running.

**Name**  CDR_PUB_EXE_RUNTIME.GetCurrentlyExecutingParams

**Signature**

FUNCTION GETCURRENTLYEXECUTINGPARAMS(
    P_API_VERSION  IN    NUMBER,
    P_INIT_MSG_LIST  IN    VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_COMMIT  IN    VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_VALIDATION_LEVEL  IN    NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL
) RETURN CDR_PARAMETER_VALUES_COLL;

**Return**  Type CDR_PARAMETER_VALUES_COLL
25.2.3 Get Information About a Job

Use this API to retrieve information about a Job by passing its JOB_ID.

**Name**  CDR_PUB_EXE_RUNTIME.GetJobInfo

**Signature**

```sql
PROCEDURE GETJOBINFO(
    P_API_VERSION  IN    NUMBER,
    P_INIT_MSG_LIST  IN    VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_COMMIT  IN    VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_VALIDATION_LEVEL  IN    NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
    X_RETURN_STATUS  OUT    VARCHAR2,
    X_MSG_COUNT  OUT    NUMBER,
    X_MSG_DATA  OUT    VARCHAR2,
    PI_NJOBID  IN    CDR_JOBS.JOB_ID%TYPE,
    PO_RSUBMISSION_V  OUT    CDR_SUBMISSIONS_V%ROWTYPE,
    PO_RJOB_V  OUT    CDR_JOBS_V%ROWTYPE,
);
```

**Parameters**  This API has standard parameters (see "Standard Parameters" on page 5) and the following parameters:

- **PI_NJOBID** (Mandatory) Enter the JOB_ID of the job you want information about. You can get the Job ID by running CDR_PUB_EXE_RUNTIME. Get Currently Executing Job ID.
- **PO_RSUBMISSION_V** This is an output parameter. The API returns the row corresponding to the SUBMISSION_ID associated with the JOB_ID value in CDR_JOBS from the CDR_SUBMISSIONS table.
- **PO_RJOB_V** This is an output parameter. The API returns the row corresponding to the JOB_ID from the CDR_JOBS table.

25.2.4 Get Job Information (Overloaded)

Use this API to retrieve information about a Job by passing its JOB_ID.

**Name**  CDR_PUB_EXE_RUNTIME.GetJobInfo

**Signature**

```sql
PROCEDURE GETJOBINFO(
    P_API_VERSION  IN    NUMBER,
    P_INIT_MSG_LIST  IN    VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_COMMIT  IN    VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_VALIDATION_LEVEL  IN    NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
    X_RETURN_STATUS  OUT    VARCHAR2,
    X_MSG_COUNT  OUT    NUMBER,
    X_MSG_DATA  OUT    VARCHAR2,
    PI_NJOBID  IN    CDR_JOBS.JOB_ID%TYPE,
    PO_RSUBMISSION  OUT    CDR_SUBMISSIONS%ROWTYPE,
    PO_RJOB  OUT    CDR_JOBS%ROWTYPE,
);
```

**Parameters**  This API has standard parameters (see "Standard Parameters" on page 5) and the following parameters:

- **PI_NJOBID** (Mandatory) Enter the JOB_ID of the job you want information about. You can get the Job ID by running CDR_PUB_EXE_RUNTIME. Get Currently Executing Job ID.
- **PO_RSUBMISSION** This is an output parameter. The API returns the row corresponding to the SUBMISSION_ID associated with the JOB_ID value in CDR_SUBMISSIONS table.
- **PO_RJOB** This is an output parameter. The API returns the row corresponding to the JOB_ID from the CDR_JOBS table.
Set Execution Statuses

This package contains APIs for setting and retrieving the execution status of Oracle LSH Programs. It also contains APIs for setting output parameters from external tools.

This section contains the following topics:

- Section 25.3.1, "Set a User-specific Completion Status"
- Section 25.3.2, "Set a Customized Output Title"
- Section 25.3.3, "Set a Customized Output Subtitle"
- Section 25.3.4, "Set an Output Parameter"
- Section 25.3.5, "Get a Completion Status"

25.3.1 Set a User-specific Completion Status

Use this API to set the completion status to a user specified value. Valid status values are: 1 for OK; 2 for OK With Warnings; 3 for Failure.

**Name**  
CDR_PUB_EXE_USER_UTILS.SetCompletionStatus

**Signature**  
PROCEDURE SETCOMPLETIONSTATUS( 
P_API_VERSION IN NUMBER,
P_INIT_MSG_LIST IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
P_COMMIT IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
P_VALIDATION_LEVEL IN NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
X_RETURN_STATUS OUT VARCHAR2,
X_MSG_COUNT OUT NUMBER,
X_MSG_DATA OUT VARCHAR2,
PI_NCOMPLETIONSTATUS IN NUMBER := 1 
);

**Parameters**  
This API has standard parameters (see "Standard Parameters" on page 5) and the following parameter:

- **PI_NJOBID** (Mandatory) Enter the JOB ID.
- **PO_RSUBMISSION** This is an output parameter. The API returns the row corresponding to the SUBMISSION_ID (associated with the JOB_ID value in CDR_JOBS) from the CDR_SUBMISSIONS table.
- **PO_RJOB** This is an output parameter. The API returns the row corresponding to the JOB_ID from the CDR_JOBS table.
- **PO_BISTOPLEVELJOB** This is an output parameter. The API returns "T" if the Job is a Top Level Job.
- **PO_BISMASTERJOB** This is an output parameter. The API returns "T" if the job is a Master job.
Set Execution Statuses

**PI_NCOMPLETIONSTATUS** (Mandatory) Enter the completion status value. Valid values are 1, 2, or 3.

### 25.3.2 Set a Customized Output Title

Use this API to set a title for individual RSE outputs.

**Name** CDR_PUB_EXE_USER_UTILS.SetCustomOutputTitle

**Signature**

```sql
PROCEDURE SETCUSTOMOUTPUTTITLE(
    P_API_VERSION IN NUMBER,
    P_INIT_MSG_LIST IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_COMMIT IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_VALIDATION_LEVEL IN NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
    X_RETURN_STATUS OUT VARCHAR2,
    X_MSG_COUNT OUT NUMBER,
    X_MSG_DATA OUT VARCHAR2,
    PI_VFILEREF IN CDR_PLANNED_OUTPUTS.FILEREF%TYPE,
    PI_VVALUE IN VARCHAR2
);
```

**Parameters** This API has standard parameters (see "Standard Parameters" on page 5) and the following parameters:

- **PI_VFILEREF** (Mandatory) Enter the file reference name of the Planned Output to which you want to assign the title.
- **PI_VVALUE** (Mandatory) Enter the title text.

### 25.3.3 Set a Customized Output Subtitle

Use this API to set a subtitle for individual RSE outputs.

**Name** CDR_PUB_EXE_USER_UTILS.SetCustomOutputSubtitle

**Signature**

```sql
PROCEDURE SETCUSTOMOUTPUTSUBTITLE(
    P_API_VERSION IN NUMBER,
    P_INIT_MSG_LIST IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_COMMIT IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_VALIDATION_LEVEL IN NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
    X_RETURN_STATUS OUT VARCHAR2,
    X_MSG_COUNT OUT NUMBER,
    X_MSG_DATA OUT VARCHAR2,
    PI_VFILEREF IN CDR_PLANNED_OUTPUTS.FILEREF%TYPE,
    PI_VVALUE IN VARCHAR2
);
```

**Parameters** This API has standard parameters (see "Standard Parameters" on page 5) and the following parameters:

- **PI_VFILEREF** (Mandatory) Enter the file reference name of the Planned Output to which you want to assign the subtitle.
- **PI_VVALUE** (Mandatory) Enter the subtitle text.
25.3.4 Set an Output Parameter
Use this API to send custom parameters and their values to Oracle LSH from external sources such as SAS. Parameter names and their values passed to this API get added to the cdr_temp_output_params table.

**Name**  
CDR_PUB_EXE_USER_UTILS.SetOutputParams

**Signature**
PROCEDURE SETOUTPUTPARAMS(  
P_API_VERSION IN NUMBER,  
P_INIT_MSG_LIST IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,  
P_COMMIT IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,  
P_VALIDATION_LEVEL IN NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,  
X_RETURN_STATUS OUT VARCHAR2,  
X_MSG_COUNT OUT NUMBER,  
X_MSG_DATA OUT VARCHAR2,  
PI_VPARAMNAME IN CDR_TEMP_OUTPUT_PARAMS.PARAMETER_NAME%TYPE,  
PI_VPARAMVALUE IN CDR_TEMP_OUTPUT_PARAMS.PARAMETER_VALUE%TYPE  
);

**Parameters**  
This API has standard parameters (see "Standard Parameters" on page 5) and the following parameters:
- **PI_VPARAMNAME** (Mandatory) Enter the parameter name.
- **PI_VPARAMVALUE** (Mandatory) Enter a value for the parameter.

25.3.5 Get a Completion Status
Use this API to retrieve a completion status value. The completion status value can only be one of the following: 1 for OK; 2 for OK With Warnings; 3 for Failure.

**Name**  
CDR_PUB_EXE_USER_UTILS.GetCompletionStatus

**Signature**
FUNCTION GETCOMPLETIONSTATUS(  
P_API_VERSION IN NUMBER,  
P_INIT_MSG_LIST IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,  
P_COMMIT IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,  
P_VALIDATION_LEVEL IN NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL  
) RETURN NUMBER;

**Return**  
Type NUMBER

**Description**  
Completion Status. The values are 1: OK 2: OK_WITH_WARNINGS 3: FAILURE

**Parameters**  
This API has standard parameters. See "Standard Parameters" on page 5) for details.

25.4 Submit Messages
This package contains one API related to the submission of messages.
25.4.1 Submit a Message

Use this API to add a submission request to the message queue.

**Name**  CDR_PUB_EXE_MSG_API.Submit_Message

**Signature**

```sql
PROCEDURE SUBMIT_MESSAGE(
    P_API_VERSION IN NUMBER,
    P_INIT_MSG_LIST IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_COMMIT IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_VALIDATION_LEVEL IN NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
    X_RETURN_STATUS OUT VARCHAR2,
    X_MSG_COUNT OUT NUMBER,
    X_MSG_DATA OUT VARCHAR2,
    PI_MSG IN VARCHAR2 := NULL
);
```

**Parameters**  This API has standard parameters (see "Standard Parameters" on page 5) and the following parameter:

- **PI_MSG**  (Mandatory). Enter in XML format, the message that you want to add to the message queue.

25.5 Create Submission Records

This package contains procedures to log messages during a job run and to create a submission record.

This section contains the following topics:

- Section 25.5.1, "Start a Job"
- Section 25.5.2, "Create a Submission"
- Section 25.5.3, "Create a Submission from a Job"
- Section 25.5.4, "Add a Job Log"

25.5.1 Start a Job

Use this API to start the execution of a job.

**Name**  CDR_PUB_EXE_SUBMISSION.StartJob

**Signature**

```sql
PROCEDURE STARTJOB(
    P_API_VERSION IN NUMBER,
    P_INIT_MSG_LIST IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_COMMIT IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_VALIDATION_LEVEL IN NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
    X_RETURN_STATUS OUT VARCHAR2,
    X_MSG_COUNT OUT NUMBER,
    X_MSG_DATA OUT VARCHAR2,
    PI_VEXECMODE IN VARCHAR2 := 'NONE',
    PI_NJOBID IN CDR_JOBS_V.JOB_ID%TYPE,
    PI_NSTREAMID IN NUMBER := NULL
);
```
Create Submission Records

Parameters
This API has standard parameters (see "Standard Parameters" on page 5) and the following parameters:

- **PI_VEXECMODE** (Mandatory) Enter the mode of execution. There are four possible values: SYNCHRONOUS (start the job and wait for completion), QUEUE (enqueue the job on the LSH queues), DIRECT (directly submit the job to OWB but do not wait), STREAM (The stream ID has to be set if this mode is used).

- **PI_NJOBID** (Mandatory) Enter the Job_ID of the job to be executed.

- **PI_NSTREAMID** Enter the OWB Stream ID. The default value is Null. This parameter is required only if mode=STREAM.

25.5.2 Create a Submission

Use this API to create a submission before starting the job.

Name
CDR_PUB_EXE_SUBMISSION.CreateSubmission

Signature
FUNCTION CREATESUBMISSION(
  P_API_VERSION IN NUMBER,
  P_INIT_MSG_LIST IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_COMMIT IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_VALIDATION_LEVEL IN NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
  PIO_OSUBMISSION IN OUT CDR_SUBMISSION_OBJ_TYPE,
  PIO_COSUBDETAILS IN OUT CDR_SUBMISSION_DETAILS_COLL,
  PI_JOBCONTEXTRC IN CDR_JOBS_V.JOB_CONTEXT_RC%TYPE,
  PI_EXECMODE IN VARCHAR2,
  PI_REFRESHTS IN CDR_JOBS_V.REFRESH_TS%TYPE
) RETURN CDR_JOBS_V.JOB_ID%TYPE;

Return
CDR_JOBS_V.JOB_ID%TYPE

Description
Returns the Job_ID. This is a number and the type is CDR_JOBS_V.JOB_ID%TYPE

Parameters
This API has standard parameters (see "Standard Parameters" on page 5) and the following parameters:

- **PIO_OSUBMISSION** (Mandatory) This parameter is of type CDR_SUBMISSION_OBJ_TYPE.

  Enter values for the following attributes: COMPANY_ID, EXECUTION_SETUP_OBJ_ID, EXECUTION_SETUP_OBJ_VER, WA_OBJ_ID, MASTER_PRREF_ID, MASTER_PRREF_VER, ACTIVE_FLAG_RC, SUBMISSION_TYPE_RC.

  You can query MASTER_PRREF_ID and MASTER_PRREF_VER from CDR_PROGRAM_REFS_V with the Object ID and version of the object to be executed as the Prref Obj Id and Prref Obj ver.

- **PIO_COSUBDETAILS** (Mandatory) This parameter is part of a collection of runtime parameters. It is of type CDR_SUBMISSION_DETAILS_COLL. Enter values for the runtime parameters that you want to include in the job submission.

  The required attributes are: COMPANY_ID, SUBMISSION_ID, PRREF_ID, PRREF_VER, PARAMETER_REF_OBJ_ID, PARAMETER_REF_OBJ_VER, PARAMETER_VALUE.

- **PI_JOBCONTEXTRC** (Mandatory) Enter the Job Context.
The possible values are the lookup_codes inside the lookup_type CDR_JOB_CONTEXTS: $JOBCONTEXT$BACKCHAIN, $JOBCONTEXT$COMPONENT, $JOBCONTEXT$SCHEDULED, $JOBCONTEXT$SUBMISSION.

- **PI_EXECMODE** (Mandatory) Enter a value for the execution mode. There are 3 possible values: SYNCHRONOUS (start the job and wait for completion), QUEUE (enqueue the job on the LSH queues), DIRECT (directly submit the job to OWB but would not wait).

- **PI_REFRESHTS** (Mandatory) Enter the Refresh timestamp you want to have associated with the job. It is normally 'sysdate,' but if you want to explicitly set a timestamp you can enter it here; for example, for a recovery job.

### 25.5.3 Create a Submission from a Job

This API creates a Submission from the initial Job.

**Name**  CDR_PUB_EXE_SUBMISSION.CreateSubmission

**Signature**

PROCEDURE CREATESUBMISSION(
    P_API_VERSION IN    NUMBER,
    P_INIT_MSG_LIST  IN    VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_COMMIT  IN    VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_VALIDATION_LEVEL  IN    NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
    X_RETURN_STATUS  OUT    VARCHAR2,
    X_MSG_COUNT  OUT    NUMBER,
    X_MSG_DATA  OUT    VARCHAR2,
    PIO_OSUBMISSION  IN OUT    CDR_SUBMISSION_OBJ_TYPE,
    PIO_COPROGRAMSUBDETAILS  IN OUT    CDR_SUBMISSION_DETAILS_COLL,
    PIO_COSYSTEMSUBDETAILS  IN OUT    CDR_SUBMISSION_DETAILS_COLL,
    PI_NOAACCOUNTID  IN    CDR_SUBMISSIONS.OA_ACCOUNT_ID%TYPE := NULL,
    PI_COSNAPSHOT  IN    CDR_SNAPSHOT_TABLE_COLL := CDR_SNAPSHOT_TABLE_COLL());

**Parameters**  This API has standard parameters (see "Standard Parameters" on page 5) and the following parameters:

- **PIO_OSUBMISSION** Object containing Submission attributes like submission_type_rc, execution_setup_id, master_prref_id etc.

- **PIO_COPROGRAMSUBDETAILS** This is part of submission details which is a collection of Run time parameters.

- **PIO_COSYSTEMSUBDETAILS** This is part of submission details which is a collection of system parameters.

- **PI_NOAACCOUNTID** Enter the OA account ID of the user making the submission.

- **PI_COSNAPSHOT** This is a collection of snapshots attributes like Tables Obj_id, Obj_Ver, source_master_job_Id etc.

### 25.5.4 Add a Job Log

Use this API to populate the cdr_job_log with the log entry for a given job.

**Name**  CDR_PUB_EXE_SUBMISSION.AddJobLogEntry

**Signature**
PROCEDURE ADDJOBLOGENTRY(
    P_API_VERSION IN NUMBER,
    P_INIT_MSG_LIST IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_COMMIT IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_VALIDATION_LEVEL IN NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
    X_RETURN_STATUS OUT VARCHAR2,
    X_MSG_COUNT OUT NUMBER,
    X_MSG_DATA OUT VARCHAR2,
    PI_NCOMPANYID IN CDR_JOB_LOG.COMPANY_ID%TYPE,
    PI_NJOBID IN CDR_JOB_LOG.JOB_ID%TYPE,
    PI_VLOGENTRY IN VARCHAR2
);

Parameters  This API has standard parameters (see “Standard Parameters” on page 5) and the following parameters:

- **PI_NCOMPANYID** Enter the Company ID.
- **PI_NJOBID** Enter the Job ID.
- **PI_VLOGENTRY** Enter a Log Entry. This accepts a text message.
This is a public interface for object security-related operations. You need the CDR_SECURITY_ADMIN or CDR_DATA_SECURITY_ADMIN application role to use any of these APIs.

26.1 Create and Modify Security Policies

This section contains the following topics:

- Section 26.1.1, "Create a Subtype"
- Section 26.1.2, "Copy a Subtype"
- Section 26.1.3, "Modify a Subtype"
- Section 26.1.4, "Assign Roles to a Subtype Operation"
- Section 26.1.5, "Assign Operations to a Subtype Role"
- Section 26.1.6, "Remove a Subtype"
- Section 26.1.7, "Create a Role"
- Section 26.1.8, "Modify a Role"
- Section 26.1.9, "Add a Group Role"
- Section 26.1.10, "Get Roles for a User"
- Section 26.1.11, "Remove a Role"
- Section 26.1.12, "Remove a Group Role"
- Section 26.1.13, "Create a User Group"
- Section 26.1.14, "Add Users to a Group"
- Section 26.1.15, "Remove Users from a Role in a User Group"
- Section 26.1.16, "Assign a User Group to an Object"
- Section 26.1.17, "Copy a User Group"
- Section 26.1.18, "Copy a User Group with its Users"
- Section 26.1.19, "Modify a User Group"
- Section 26.1.20, "Remove All Group Roles from a User Group"
- Section 26.1.21, "Remove All Users in a Group"
- Section 26.1.22, "Revoke a User Group From an Object"
- Section 26.1.23, "Undo a Revoke a User Group Action"
26.1.24, "Remove a User Group"
26.1.25, "Unassign a User Group From an Object"
26.1.26, "Unassign Roles from an Operation on an Object's Subtype"
26.1.27, "Unassign Operations on an Object Subtype's Role"

26.1.1 Create a Subtype
Use this API to create a new subtype.

**Name**  CDR_PUB_SECURITY_PKG.CreateSubtype

**Signature**

PROCEDURE CREATE_SUBTYPE(
    P_API_VERSION IN NUMBER,
    P_INIT_MSG_LIST IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_COMMIT IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_VALIDATION_LEVEL IN NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
    X_RETURN_STATUS OUT VARCHAR2,
    X_MSG_COUNT OUT NUMBER,
    X_MSG_DATA OUT VARCHAR2,
    PIO_SUBTYPE IN OUT CDR_SUBTYPE_OBJ_TYPE
);

**Parameters**  This API has standard parameters (see "Standard Parameters" on page 5) and the following parameter:

**PIO_SUBTYPE** (Mandatory) This is a collection of CDR_USER_GROUPS_OBJ_TYPE that contain attributes related to User Groups.

The required attributes are: NAME,OBJECT_SUBTYPE_ID,OBJECT_TYPE_RC.

26.1.2 Copy a Subtype
Use this API to make a copy of a subtype.

**Name**  CDR_PUB_SECURITY_PKG.CopySubtype

**Signature**

PROCEDURE COPY_SUBTYPE(
    P_API_VERSION IN NUMBER,
    P_INIT_MSG_LIST IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_COMMIT IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_VALIDATION_LEVEL IN NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
    X_RETURN_STATUS OUT VARCHAR2,
    X_MSG_COUNT OUT NUMBER,
    X_MSG_DATA OUT VARCHAR2,
    PI_SUBTYPE IN CDR_SUBTYPE_OBJ_TYPE
);

**Parameters**  This API has standard parameters (see "Standard Parameters" on page 5) and the following parameter:

**PI_SUBTYPE** (Mandatory) This a parameter of table type CDR_SUBTYPE_OBJ_TYPE that contains information about the subtype.
26.1.3 Modify a Subtype

Use this API to update a subtype.

**Name**  CDR_PUB_SECURITY_PKG.ModifySubtype

**Signature**

PROCEDURE MODIFYSUBTYPE(
    P_API_VERSION  IN    NUMBER,
    P_INIT_MSG_LIST  IN    VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_COMMIT  IN    VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_VALIDATION_LEVEL  IN    NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
    X_RETURN_STATUS  OUT    VARCHAR2,
    X_MSG_COUNT  OUT    NUMBER,
    X_MSG_DATA  OUT    VARCHAR2,
    PIO_SUBTYPE  IN OUT    CDR_SUBTYPE_OBJ_TYPE
);

**Parameters**  This API has standard parameters (see “Standard Parameters” on page 5) and the following parameter:

PIO_SUBTYPE  (Mandatory) This is a parameter of table type CDR_SUBTYPE_OBJ_TYPE that contains information about the object subtype.

26.1.4 Assign Roles to a Subtype Operation

Use this API to assign a role to an operation for a subtype of an object.

**Name**  CDR_PUB_SECURITY_PKG.AssignRolesToSubtypeOperation

**Signature**

PROCEDURE ASSIGNROLESTOSUBTYPEOPERATION(
    P_API_VERSION  IN    NUMBER,
    P_INIT_MSG_LIST  IN    VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_COMMIT  IN    VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_VALIDATION_LEVEL  IN    NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
    X_RETURN_STATUS  OUT    VARCHAR2,
    X_MSG_COUNT  OUT    NUMBER,
    X_MSG_DATA  OUT    VARCHAR2,
    PIO_ROLESTOOPR  IN OUT    CDR_SUBTYPE_OPR_ROLES_COLL
);

**Parameters**  This API has standard parameters (see “Standard Parameters” on page 5) and the following parameters:

PIO_ROLESTOOPR  (Mandatory) This is a collection of CDR_SUBTYPE_OPR_ROLE_OBJ_TYPE.

26.1.5 Assign Operations to a Subtype Role

Use this API to assign an operation to a role of an object subtype.

**Name**  CDR_PUB_SECURITY_PKG.AssignOprToSubtypeRole

**Signature**

PROCEDURE ASSIGNOPRTOSUBTYPEROLE(
    P_API_VERSION  IN    NUMBER,
Create and Modify Security Policies

This API has standard parameters (see "Standard Parameters" on page 5) and the following parameter:

PIO_ROLESTOOPR (Mandatory) This is a collection of CDR_SUBTYPE_OPR_ROLE_OBJ_TYPE.

26.1.6 Remove a Subtype

Use this API to delete a subtype that is not Active. If objects are assigned to the subtype, you cannot delete it even if it is Inactive.

Name CDR_PUB_SECURITY_PKG.RemoveSubtype

Signature

PROCEDURE REMOVESUBTYPE(
P_API_VERSION IN NUMBER,
P_INIT_MSG_LIST IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
P_COMMIT IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
P_VALIDATION_LEVEL IN NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
X_RETURN_STATUS OUT VARCHAR2,
X_MSG_COUNT OUT NUMBER,
X_MSG_DATA OUT VARCHAR2,
PI_SUBTYPEID IN CDR_OBJECT_SUBTYPES_B.OBJECT_SUBTYPE_ID%TYPE,
PI_COMPANYID IN CDR_OBJECT_SUBTYPES_B.COMPANY_ID%TYPE,
PI_OBJECTTYPERC IN CDR_OBJECT_SUBTYPES_B.OBJECT_TYPE_RC%TYPE
)

Parameters This API has standard parameters (see "Standard Parameters" on page 5) and the following parameters:

- PI_SUBTYPEID (Mandatory) Enter the OBJECT_SUBTYPE_ID of the subtype you want to delete.
- PI_COMPANYID (Mandatory) Enter the COMPANY_ID associated with the OBJ_SUBTYPE_ID.
- PI_OBJECTTYPERC (Mandatory) Enter the OBJECT_TYPE_RC value for the object type associated with the subtype.

26.1.7 Create a Role

Use this API to create a new Role.

Name CDR_PUB_SECURITY_PKG.CreateRole

Signature

PROCEDURE CREATEROLE(
P_API_VERSION IN NUMBER,
P_INIT_MSG_LIST IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
26.1.8 Modify a Role

Use this API to update a Role. You can change the name, description, and Active status of a Role.

**Name**  CDR_PUB_SECURITY_PKG.ModifyRole

**Signature**

PROCEDURE MODIFYROLE(
  P_API_VERSION IN NUMBER,
  P_INIT_MSG_LIST IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_COMMIT IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_VALIDATION_LEVEL IN NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
  X_RETURN_STATUS OUT VARCHAR2,
  X_MSG_COUNT OUT NUMBER,
  X_MSG_DATA OUT VARCHAR2,
  PIO_ROLE IN OUT CDR_ROLE_OBJ_TYPE
);

**Parameters**  This API has standard parameters (see “Standard Parameters” on page 5) and the following parameters:

PIO_ROLE (Mandatory) This is a parameter of table type CDR_ROLE_OBJ_TYPE that contains attributes related to a Role.

Required Attributes are: NAME, CODE, OBJECT VERSION NUMBER (pass 1 for this).

26.1.9 Add a Group Role

Use this API to create roles for a User Group.

**Name**  CDR_PUB_SECURITY_PKG.AddGrpRoles

**Signature**

PROCEDURE ADDGRPROLES(
  P_API_VERSION IN NUMBER,
  P_INIT_MSG_LIST IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_COMMIT IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_VALIDATION_LEVEL IN NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
  X_RETURN_STATUS OUT VARCHAR2,
  X_MSG_COUNT OUT NUMBER,
  X_MSG_DATA OUT VARCHAR2,
  PIO_ROLES IN OUT CDR_UG_ROLE_OBJ_TYPE
);
Create and Modify Security Policies

26.1.10 Get Roles for a User
Use this API to retrieve all Roles assigned to a user.

**Name**  CDR_PUB_SECURITY_PKG.GetRolesForUser

**Signature**

FUNCTION GETROLESFORUSER (  
  P_API_VERSION IN NUMBER,  
  P_INIT_MSG_LIST IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,  
  P_COMMIT IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,  
  P_VALIDATION_LEVEL IN NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,  
  P_USERID IN VARCHAR2  
) RETURN CLOB;

**Return**  Type CLOB

Description CLOB for all roles for given user.

**Parameters**  This API has standard parameters. See “Standard Parameters” on page 5) for details.

26.11 Remove a Role
Use this API to delete a role.

**Name**  CDR_PUB_SECURITY_PKG.RemoveRole

**Signature**

PROCEDURE REMOVEROLE (  
  P_API_VERSION IN NUMBER,  
  P_INIT_MSG_LIST IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,  
  P_COMMIT IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,  
  P_VALIDATION_LEVEL IN NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,  
  X_RETURN_STATUS OUT VARCHAR2,  
  X_MSG_COUNT OUT NUMBER,  
  X_MSG_DATA OUT VARCHAR2,  
  PI_COMPANYID IN NUMBER,  
  PI_ROLEID IN NUMBER  
);

**Parameters**  This API has standard parameters (see “Standard Parameters” on page 5) and the following parameters:

- **PI_COMPANYID** (Mandatory) Enter the COMPANY_ID associated with the Role.
- **PI_ROLEID** (Mandatory) Enter the ROLE_ID of the Role that you want to delete.
26.1.12 Remove a Group Role

Use this API to remove a single Role from a User Group. You can remove all Roles from the User Group at the same time by using the Remove All Group Roles API.

**Name**  CDR_PUB_SECURITY_PKG.RemoveGrpRoles

**Signature**

PROCEDURE REMOVEGRPROLES(
    P_API_VERSION IN NUMBER,
    P_INIT_MSG_LIST IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_COMMIT IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_VALIDATION_LEVEL IN NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
    X_RETURN_STATUS OUT VARCHAR2,
    X_MSG_COUNT OUT NUMBER,
    X_MSG_DATA OUT VARCHAR2,
    PIO_ROLES IN CDR_UG_ROLE_OBJ_TYPE
);  

**Parameters**  This API has standard parameters (see “Standard Parameters” on page 5) and the following parameter:

**PIO_ROLES**  (Mandatory) This parameter is of table type CDR_UG_ROLE_OBJ_TYPE that contains information about User Groups and Roles.

26.1.13 Create a User Group

Use this API to create a new User Group.

**Name**  CDR_PUB_SECURITY_PKG.CreateUserGroup

**Signature**

PROCEDURE CREATEUSERGROUP(
    P_API_VERSION IN NUMBER,
    P_INIT_MSG_LIST IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_COMMIT IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_VALIDATION_LEVEL IN NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
    X_RETURN_STATUS OUT VARCHAR2,
    X_MSG_COUNT OUT NUMBER,
    X_MSG_DATA OUT VARCHAR2,
    PIO_USERGRP IN OUT CDR_USER_GROUP_OBJ_TYPE
);  

**Parameters**  This API has standard parameters (see “Standard Parameters” on page 5) and the following parameters:

**PIO_USERGRP**  (Mandatory) This is a collection of CDR_USER_GROUPS_OBJ_TYPE that contains attributes related to User Groups.

Required attributes are: USER_GROUP_ID,COMPANY_ID, NAME.

26.1.14 Add Users to a Group

Use this API to add users to a User Group.

**Name**  CDR_PUB_SECURITY_PKG.AddUserToGrp

**Signature**
PROCEDURE ADDUSERTOGRP(
    P_API_VERSION IN NUMBER,
    P_INIT_MSG_LIST IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_COMMIT IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_VALIDATION_LEVEL IN NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
    X_RETURN_STATUS OUT VARCHAR2,
    X_MSG_COUNT OUT NUMBER,
    X_MSG_DATA OUT VARCHAR2,
    PIO_USERUGROLES IN OUT CDR_USER_UG_ROLE_OBJ_TYPE
);  

Parameters  This API has standard parameters (see "Standard Parameters" on page 5) and the following parameters:  

PIO_USERUGROLES (Mandatory) This is a parameter of table type CDR_USER_UG_ROLE_OBJ_TYPE.

Required Attributes are: UG_COMPANY_ID, USER_GROUP_ID, ROLE_ID, USER_ID, ROLE_ID

26.1.15 Remove Users from a Role in a User Group

Use this API to delete users from a Role in a User Group.

Name  CDR_PUB_SECURITY_PKG.RemoveUsersInGrp

Signature

PROCEDURE REMOVEUSERSINGRP(
    P_API_VERSION IN NUMBER,
    P_INIT_MSG_LIST IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_COMMIT IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_VALIDATION_LEVEL IN NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
    X_RETURN_STATUS OUT VARCHAR2,
    X_MSG_COUNT OUT NUMBER,
    X_MSG_DATA OUT VARCHAR2,
    PIO_USERUGROLES IN OUT CDR_USER_UG_ROLE_OBJ_TYPE
);  

Parameters  This API has standard parameters (see "Standard Parameters" on page 5) and the following parameter:  

PIO_USERUGROLES (Mandatory) This parameter is of table type CDR_USER_UG_ROLE_OBJ_TYPE that contains information about user, User Groups, and Roles.

26.1.16 Assign a User Group to an Object

Use this API to assign a User Group to an object.

Name  CDR_PUB_SECURITY_PKG.AssignUsrGrpToObj

Signature

PROCEDURE ASSIGNUSRGRPTOOBJ(
    P_API_VERSION IN NUMBER,
    P_INIT_MSG_LIST IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_COMMIT IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_VALIDATION_LEVEL IN NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
    X_RETURN_STATUS OUT VARCHAR2,
    X_MSG_COUNT OUT NUMBER,
Parameters

This API has standard parameters (see “Standard Parameters” on page 5) and the following parameters:

- **PI_BASEOBJECTTYPE** (Mandatory) This is a parameter of table type CDR_BASE_OBJ_TYPE that contains object attributes.
  
  Provide the basic naming attributes for the object to which you want to assign the User Group. (COMPANY_ID, OBJECT_ID, OBJECT_VER, NAMESPACE_OBJ_ID, NAMESPACE_OBJ_VER, OBJECT_VERSION_NUMBER). Initialize these attributes in CDR_BASE_OBJ_TYPE.

- **PI_CDROBJUGCOLL** (Mandatory) This is a collection of CDR_OBJ_UG_OBJ_TYPE. Enter User Group details in this parameter.
  
  The following are required parameters: UG_COMPANY_ID, OBJ_COMPANY_ID, USER_GROUP_ID, OBJ_ID and EXCLUSION_FLAG.

### 26.1.17 Copy a User Group

Use this API to make a copy of a User Group including its Roles but not its users.

**Name**  
CDR_PUB_SECURITY_PKG.CopyUserGroup

**Signature**

PROCEDURE COPYUSERGROUP(
  P_API_VERSION IN NUMBER,
  P_INIT_MSG_LIST IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_COMMIT IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_VALIDATION_LEVEL IN NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
  X_RETURN_STATUS OUT VARCHAR2,
  X_MSG_COUNT OUT NUMBER,
  X_MSG_DATA OUT VARCHAR2,
  PIO_USERGRP IN OUT CDR_USER_GROUPS_COLL
);
P_VALIDATION_LEVEL IN NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
X_RETURN_STATUS OUT VARCHAR2,
X_MSG_COUNT OUT NUMBER,
X_MSG_DATA OUT VARCHAR2,
PIO_USERGRP IN OUT CDR_USER_GROUPS_COLL
);

Parameters This API has standard parameters (see "Standard Parameters" on page 5) and the following parameter:

PIO_USERGRP (Mandatory) This is a collection of CDR_USER_GROUPS_OBJ_TYPE that contains attributes related to User Groups.

Required attributes are: USER_GROUP_ID,COMPANY_ID, NAME.

26.1.19 Modify a User Group

Use this API to modify a User Group.

Name CDR_PUB_SECURITY_PKG.ModifyUserGroup

Signature

PROCEDURE MODIFYUSERGROUP(
P_API_VERSION IN NUMBER,
P_INIT_MSG_LIST IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
P_COMMIT IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
P_VALIDATION_LEVEL IN NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
X_RETURN_STATUS OUT VARCHAR2,
X_MSG_COUNT OUT NUMBER,
X_MSG_DATA OUT VARCHAR2,
PIO_USERGRP IN OUT CDR_USER_GROUP_OBJ_TYPE
);

Parameters This API has standard parameters (see "Standard Parameters" on page 5) and the following parameters:

PIO_USERGRP (Mandatory) This is a collection of CDR_USER_GROUPS_OBJ_TYPE that contains attributes related to User Groups.

Required attributes are: USER_GROUP_ID,COMPANY_ID, NAME.

26.1.20 Remove All Group Roles from a User Group

Use this API to remove all Roles from the User Group.

Name CDR_PUB_SECURITY_PKG.RemoveAllGrpRoles

Signature

PROCEDURE REMOVEALLGRPROLES(
P_API_VERSION IN NUMBER,
P_INIT_MSG_LIST IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
P_COMMIT IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
P_VALIDATION_LEVEL IN NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
X_RETURN_STATUS OUT VARCHAR2,
X_MSG_COUNT OUT NUMBER,
X_MSG_DATA OUT VARCHAR2,
PI_USERGRPID IN NUMBER,
PI_COMPANY_ID IN NUMBER
);
Parameters  This API has standard parameters (see "Standard Parameters" on page 5) and the following parameters:

- **PI_USERGRPID** (Mandatory) Enter the numeric ID to identify the User Group. This parameter is of type Number and corresponds to the CDR_UG_ROLES.USER_GROUP_ID%TYPE column.
- **PI_COMPANY_ID** (Mandatory) Enter the Company Id.

### 26.1.21 Remove All Users in a Group

Use this API to remove all users from a Role in a User Group.

**Name**  CDR_PUB_SECURITY_PKG.RemoveAllUsersInGrp

**Signature**

```sql
PROCEDURE REMOVEALLUSERSINGRP(
    P_API_VERSION  IN    NUMBER,
    P_INIT_MSG_LIST  IN    VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_COMMIT  IN    VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_VALIDATION_LEVEL  IN    NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
    X_RETURN_STATUS  OUT    VARCHAR2,
    X_MSG_COUNT  OUT    NUMBER,
    X_MSG_DATA  OUT    VARCHAR2,
    PIO_USERUGROLES  IN OUT    CDR_USER_UG_ROLE_OBJ_TYPE
);
```

Parameters  This API has standard parameters (see "Standard Parameters" on page 5) and the following parameter:

- **PIO_USERUGROLES** (Mandatory) This parameter is of table type CDR_USER_UG_ROLE_OBJ_TYPE that contains information about users, User Groups, and Roles.

### 26.1.22 Revoke a User Group From an Object

Use this API to revoke a User Group from an object.

To remove access to an object through an inherited User Group, you must revoke the User Group assignment.

Use Unassign User Group from Object (UNASSIGNUSRGRPFROMOBJ) for User Groups assigned explicitly.

**Name**  CDR_PUB_SECURITY_PKG.RevokeUsrGrpFromObJ

**Signature**

```sql
PROCEDURE REVOKEUSRGRPFROMOBJ(
    P_API_VERSION  IN    NUMBER,
    P_INIT_MSG_LIST  IN    VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_COMMIT  IN    VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_VALIDATION_LEVEL  IN    NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
    X_RETURN_STATUS  OUT    VARCHAR2,
    X_MSG_COUNT  OUT    NUMBER,
    X_MSG_DATA  OUT    VARCHAR2,
    PI_BASEOBJECTTYPE  IN OUT    CDR_BASE_OBJ_TYPE,
    PI_CDROBJUGOBJTYPE  IN    CDR_OBJ_UG_OBJ_TYPE,
    PO_HASVIEWPERMAFTERREVOKE  OUT    VARCHAR2
);
```
Create and Modify Security Policies

Parameters This API has standard parameters (see "Standard Parameters" on page 5) and the following parameters:

- **PI_BASEOBJECTTYPE** (Mandatory) This is a parameter of table type CDR_BASE_OBJ_TYPE that contains object attributes. Enter values to identify the object from which the User Group is to be revoked.

  The following attributes are required: COMPANY_ID,OBJ_ID,OBJ_VER,OBJECT_VERSION_NUMBER,NAMESPACE_OBJ_ID,NAMESPACE_OBJ_VER.

- **PI_CDROBJUGOBJTYPE** (Mandatory) This is a parameter of table type CDR_OBJ_UG_OBJ_TYPE that contains information about the object and the User Group.

- **PO_HASVIEWPERMAFTERREVOKE** (Mandatory) Enter appropriate values for this parameter to specify whether view permissions exist after the revoking of the User Group from the object.

26.1.23 Undo a Revoke a User Group Action

Use this API to undo the revoking of a User Group from an object.

Name  CDR_PUB_SECURITY_PKG.UnrevokeUsrGrpFromObj

Signature

PROCEDURE UNREVOKEUSRGRPFROMOBJ(
  P_API_VERSION IN NUMBER,
  P_INIT_MSG_LIST IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_COMMIT IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_VALIDATION_LEVEL IN NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
  X_RETURN_STATUS OUT VARCHAR2,
  X_MSG_COUNT OUT NUMBER,
  X_MSG_DATA OUT VARCHAR2,
  PI_BASEOBJECTTYPE IN OUT CDR_BASE_OBJ_TYPE,
  PI_CDROBJUGOBJTYPE IN CDR_OBJ_UG_OBJ_TYPE
);

Parameters This API has standard parameters (see "Standard Parameters" on page 5) and the following parameters:

- **PI_BASEOBJECTTYPE** (Mandatory) This is a parameter of table type CDR_BASE_OBJ_TYPE that contains object attributes. Enter values to identify the object.

  The following attributes are required: COMPANY_ID,OBJ_ID,OBJ_VER,OBJECT_VERSION_NUMBER,NAMESPACE_OBJ_ID,NAMESPACE_OBJ_VER.

- **PI_CDROBJUGOBJTYPE** (Mandatory) This is a parameter of table type CDR_OBJ_UG_OBJ_TYPE that contains information about the object and the User Group.

26.1.24 Remove a User Group

Use this API to delete a User Group from the system. Once deleted, a User Group cannot be reactivated.

Name  CDR_PUB_SECURITY_PKG.RemoveUserGroup

Signature

PROCEDURE REMOVEUSERGROUP (}
26.1.25 Unassign a User Group From an Object

Use this API to unassign a User Group from an object. You can unassign User Groups explicitly assigned to the Object. You have to revoke User Groups that are inherited.

**Name**  CDR_PUB_SECURITY_PKG.UnassignUsrGrpFromObj

**Signature**

PROCEDURE UNASSIGNUSRGRPFROMOBJ(
  P_API_VERSION  IN    NUMBER,
  P_INIT_MSG_LIST  IN    VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_COMMIT  IN    VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_VALIDATION_LEVEL  IN    NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
  X_RETURN_STATUS  OUT    VARCHAR2,
  X_MSG_COUNT  OUT    NUMBER,
  X_MSG_DATA  OUT    VARCHAR2,
  PI_BASEOBJECTTYPE  IN OUT    CDR_BASE_OBJ_TYPE,
  PI_CDROBJUGOBJTYPE  IN    CDR_OBJ_UG_OBJ_TYPE,
  PO_HASVIEWPERMAFTERUNASSIGN  OUT    VARCHAR2
);

**Parameters**  This API has standard parameters (see "Standard Parameters" on page 5) and the following parameters:

- **PI_BASEOBJECTTYPE** (Mandatory) This is a parameter of table type CDR_BASE_OBJ_TYPE that contains object attributes. Enter values to identify the object.
  
  The required attributes are COMPANY_ID,OBJ_ID,OBJ_VER,OBJECT_VERSION_NUMBER,NAMESPACE_OBJ_ID,NAMESPACE_OBJ_VER.

- **PI_CDROBJUGOBJTYPE** (Mandatory) This is a parameter of table type CDR_OBJ_UG_OBJ_TYPE that contains information about the object and the User Group.

- **PO_HASVIEWPERMAFTERUNASSIGN** (Mandatory) Enter appropriate values for this parameter to specify whether view permissions exist after unassigning the User Group from the object.

26.1.26 Unassign Roles from an Operation on an Object's Subtype

Use this API to unassign Roles from an Operation on an object's subtype.
**Name**  CDR_PUB_SECURITY_PKG.UnassignRoleToSubtypeOperation

**Signature**

PROCEDURE UNASSIGNROLETOSUBTYPEOPERATION(
    P_API_VERSION IN NUMBER,
    P_INIT_MSG_LIST IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_COMMIT IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_VALIDATION_LEVEL IN NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
    X_RETURN_STATUS OUT VARCHAR2,
    X_MSG_COUNT OUT NUMBER,
    X_MSG_DATA OUT VARCHAR2,
    PI_STOPROLE IN OUT CDR_SUBTYPE_OPR_ROLE_OBJ_TYPE
);

**Parameters**  This API has standard parameters (see "Standard Parameters" on page 5) and the following parameter:

PI_STOPROLE  (Mandatory) This parameter is of table type CDR_SUBTYPE_OPR_ROLE_OBJ_TYPE that contains information about object subtype, Role, and operation.

### 26.1.27 Unassign Operations on an Object Subtype’s Role

Use this API to unassign operations on an object subtype’s role.

**Name**  CDR_PUB_SECURITY_PKG.UnassignOprToSubtypeRole

**Signature**

PROCEDURE UNASSIGNOPRTOSUBTYPEROLE(
    P_API_VERSION IN NUMBER,
    P_INIT_MSG_LIST IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_COMMIT IN VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
    P_VALIDATION_LEVEL IN NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
    X_RETURN_STATUS OUT VARCHAR2,
    X_MSG_COUNT OUT NUMBER,
    X_MSG_DATA OUT VARCHAR2,
    PI_STOPROLE IN OUT CDR_SUBTYPE_OPR_ROLE_OBJ_TYPE
);

**Parameters**  This API has standard parameters (see "Standard Parameters" on page 5) and the following parameter:

PI_STOPROLE  (Mandatory) This parameter is of table type CDR_SUBTYPE_OPR_ROLE_OBJ_TYPE that contains information about object subtype, Role, and operation.
This is a public interface for validation-related operations.

This section contains the following topics:

- Section 27.1, "Validate Objects"
- Section 27.2, "Create and Modify Validation Supporting Documents"

## 27.1 Validate Objects

This section contains one API for updating the validation status of an object.

### 27.1.1 Update an Object's Validation Status

Use this API to update an object's validation status. The API performs a cascade validation on this object and its related objects.

If this object is an instance, the API also validates its source definition. If this object contains other objects with a validation status, the API updates the validation status of all of them; and if they are instances, their source definitions.

The operation fails if any of the underlying definitions are checked out. If you are validating a Report Set, the operation also fails if any of the Program instances in the Report Set have a validation status lower than the one to which the Report Set is being upgraded.

**Name**  
CDR_PUB_VL_VALIDATION.UpdateValStatus

**Signature**

```
PROCEDURE UPDATEVALSTATUS(
  P_API_VERSION IN    NUMBER,
  P_INIT_MSG_LIST IN    VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_COMMIT IN    VARCHAR2 := CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_VALIDATION_LEVEL IN    NUMBER := CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
  X_RETURN_STATUS OUT    VARCHAR2,
  X_MSG_COUNT OUT    NUMBER,
  X_MSG_DATA OUT    VARCHAR2,
  PI_VALOBJ IN    CDR_VAL_STATUS_OBJ_TYPE,
  PO_CASCADEOBJCOLL OUT    CDR_BASE_OBJ_COLL,
  PO_ERRORNAMINGCOLL OUT    CDR_BASE_OBJ_COLL
);
```

**Parameters**  
This API has standard parameters (see "Standard Parameters" on page 5) and the following parameters:
Create and Modify Validation Supporting Documents

- **PI_VALOBJ** (Mandatory) This is a parameter of table type CDR_VAL_STATUS_OBJ_TYPE that contains object attributes. Enter values to identify the object whose validation status you want to update.

  The following attributes are required: COMPANY_ID, OBJ_ID, OBJ_VER, OBJ_TYPE_RC, VALIDATION_STATUS_RC, OBJECT_VERSION_NUMBER.

- **PO_CASCADEDOBJCOLL** This output parameter is a collection of all the objects whose validation status was updated due to cascading. If this parameter contains a value, the validation update operation succeeded.

- **PO_ERRORNAMINGCOLL** This output parameter is a collection of objects whose validation status could not be updated. If this parameter contains a value, the validation update operation failed.

### 27.2 Create and Modify Validation Supporting Documents

This section contains the following topics:

- **Section 27.2.1, ”Create a Validation Supporting Document”**
- **Section 27.2.2, ”Update a Validation Supporting Document”**
- **Section 27.2.3, ”Obsolete a Validation Supporting Document”**

#### 27.2.1 Create a Validation Supporting Document

Use this API to create a validation supporting document.

**Name** CDR_PUB_VL_VALIDATION.CreateValDocument

**Signature**

PROCEDURE CREATEVALDOCUMENT(
P_API_VERSION IN NUMBER,
P_INIT_MSG_LIST IN VARCHAR2 DEFAULT CDR_PUB_DEF_CONSTANTS.G_FALSE,
P_COMMIT IN VARCHAR2 DEFAULT CDR_PUB_DEF_CONSTANTS.G_FALSE,
P_VALIDATION_LEVEL IN NUMBER DEFAULT CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
X_RETURN_STATUS OUT NOCOPY VARCHAR2,
X_MSG_COUNT OUT NOCOPY NUMBER,
X_MSG_DATA OUT NOCOPY VARCHAR2,
PI_VALDOCOBJ IN CDR_VAL_DOC_BLOB_OBJ_TYPE)
;

**Parameters** This API has standard parameters (see ”Standard Parameters” on page 5) and the following parameter:

**PI_VALDOCOBJ** (Mandatory) This is a parameter of type CDR_VAL_DOC_BLOB_OBJ_TYPE that contains object attributes.

The following attributes are required: COMPANY_ID, OBJ_ID, OBJ_VER, DOCUMENT_NAME, DESCRIPTION, FILE_NAME, OS_FILE_PATH, FILE_BLOB, FILE_CONTENT_TYPE.

#### 27.2.2 Update a Validation Supporting Document

You can use this API to upload a new document, change attributes such as its description, or both.

**Name** CDR_PUB_VL_VALIDATION.UpdateValDocument
### Signature

```sql
PROCEDURE UPDATEVALDOCUMENT(
  P_API_VERSION IN NUMBER,
  P_INIT_MSG_LIST IN VARCHAR2 DEFAULT CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_COMMIT IN VARCHAR2 DEFAULT CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_VALIDATION_LEVEL IN NUMBER DEFAULT CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
  X_RETURN_STATUS OUT NOCOPY VARCHAR2,
  X_MSG_COUNT OUT NOCOPY NUMBER,
  X_MSG_DATA OUT NOCOPY VARCHAR2,
  PI_VALDOCOBJ IN CDR_VAL_DOC_BLOB_OBJ_TYPE);
```

**Parameters** This API has standard parameters (see “Standard Parameters” on page 5) and the following parameter:

- **PI_VALDOCOBJ** (Mandatory) This is a parameter of type CDR_VAL_DOC_BLOB_OBJ_TYPE that contains object attributes. The following attributes of the document to be updated are required: DOCUMENT_ID, COMPANY_ID, OBJ_ID, OBJ_VER, DOCUMENT_NAME, DOC_STATUS_RC, DOCUMENT_VER, DESCRIPTION, CHANGE_REASON, FILE_ID, FILE_NAME, OS_FILE_PATH, FILE_BLOB, FILE_CONTENT_TYPE, OBJECT_VERSION_NUMBER.

To get the OBJECT_VERSION_NUMBER, enter the following query:

```sql
select Max(OBJECT_VERSION_NUMBER) from cdr_vl_val_docs_v
where OBJ_ID = <objid> and OBJ_VER = <objver> and DOC_STATUS_RC = 'SVALINFOSTATUSACTIVE';
```

### 27.2.3 Obsolete a Validation Supporting Document

Use this API to remove a validation supporting document.

**Name** CDR_PUB_VL_VALIDATION.RemoveValDocument

### Signature

```sql
PROCEDURE REMOVEVALDOCUMENT(
  P_API_VERSION IN NUMBER,
  P_INIT_MSG_LIST IN VARCHAR2 DEFAULT CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_COMMIT IN VARCHAR2 DEFAULT CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_VALIDATION_LEVEL IN NUMBER DEFAULT CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
  X_RETURN_STATUS OUT NOCOPY VARCHAR2,
  X_MSG_COUNT OUT NOCOPY NUMBER,
  X_MSG_DATA OUT NOCOPY VARCHAR2,
  PI_VALDOCOBJ IN CDR_VAL_DOC_BLOB_OBJ_TYPE);
```

**Parameters** This API has standard parameters (see “Standard Parameters” on page 5) and the following parameter:

- **PI_VALDOCOBJ** (Mandatory) This is a parameter of type CDR_VAL_DOC_BLOB_OBJ_TYPE that contains object attributes.

The following attributes of the document to be obsoleted are required: DOCUMENT_ID, COMPANY_ID, DOCUMENT_VER, OBJECT_VERSION_NUMBER.

To get the OBJECT_VERSION_NUMBER, enter the following query:

```sql
select Max(OBJECT_VERSION_NUMBER) from cdr_vl_val_docs_v
where OBJ_ID = <objid> and OBJ_VER = <objver> and DOC_STATUS_RC = 'SVALINFOSTATUSACTIVE';
```
This part of the Oracle Life Sciences Data Hub (Oracle LSH) API guide contains public APIs included in Oracle Health Sciences Data Management Workbench (Oracle DMW). You can call these APIs from custom programs used in Oracle DMW.

To use these APIs you need to understand defined objects in Oracle LSH. Refer to Part I, "Essential Information" to know more about running APIs in Oracle LSH. Part 1 includes Reference Information, which contains details about the Standard Parameters required to call APIs.

Setup Utilities contains information on utility APIs that you can use to get the information and IDs you need, to invoke Oracle DMW APIs.

This part includes the following chapters:

- Chapter 28, "Introduction to Oracle DMW APIs"
- Chapter 29, "Flags, Models and Actions"
- Chapter 30, "Code Lists"
- Chapter 31, "Validation Checks"
- Chapter 32, "Transformations"
Introduction to Oracle DMW APIs

Oracle Health Sciences Data Management Workbench (Oracle DMW) is built on top of Oracle Life Sciences Data Hub (Oracle LSH) and shares the same database and execution engine. Be sure to read the following sections on Oracle LSH APIs:

- Chapter 1, "Using Application Programming Interfaces"
- Chapter 2, "Reference Information"
- Chapter 19, "Setup Utilities"

For information on Oracle DMW functionality and structure, see the Oracle Health Sciences Data Management Workbench User’s Guide, including information on object ownership in the Reference Information appendix.

**Note:** Even though an Oracle DMW Study is an Oracle LSH domain, it has additional attributes and you cannot create a study using the API for creating a domain. The API for creating a study is not public in Release 2.3.1.

**Note:** During its initial development, Oracle DMW was known as DME. Therefore many internal names contain the string dme. Please think of DME as a synonym for DMW—just as CDR was the early name for LSH.

This section includes:

- Section 28.1, "Set Up Study Environment"
- Section 28.2, "Create or Modify an Expression"
- Section 28.3, "Using APIs to Create Custom Programs"

### 28.1 Set Up Study Environment

The program you are writing must call DME_PUB_INITIALIZATION.SetupAPIStudyEnvironment before it calls any other Oracle DMW API or uses any public view. This procedure checks that the study ID and lifecycle value you pass in are valid and then uses those values and the study’s partition ID for the lifecycle to set SYS_CONTEXT appropriately. Call it again to change the study or lifecycle context.
DME_PUB_INITIALIZATION.SetupAPIStudyEnvironment also calls the Oracle LSH security package cdr_pub_api_initialization; see Section 1.2.2, "Calling the Security API Package".

---

**Note:** Like CDR_PUB_API_INITIALIZATION, access to DME_PUB_INITIALIZATION must be granted by an administrator to each individual user to start using the external APIs.

---

### 28.1.1 Initialize a Study and Lifecycle

**Name**  
DME_PUB_INITIALIZATION.SetupAPIStudyEnvironment

**Signature:**

```plaintext
PROCEDURE SETUPAPISTUDYENVIRONMENT,
( P_API_VERSION IN NUMBER,
  P_INIT_MSG_LIST IN VARCHAR2 DEFAULT CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_COMMIT IN VARCHAR2 DEFAULT CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_VALIDATION_LEVEL IN NUMBER DEFAULT CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
  X_RETURN_STATUS OUT NOCOPY VARCHAR2,
  X_MSG_COUNT OUT NOCOPY NUMBER,
  X_MSG_DATA OUT NOCOPY VARCHAR2,
  PI_COMPANY_ID IN NUMBER,
  PI_STUDY_ID IN NUMBER,
  PI_LIFECYCLE IN VARCHAR2,
);```

**Parameters:** This API has standard parameters (see "Standard Parameters" on page 2-5) as well as the following parameters:

- **PI_COMPANY_ID** (Mandatory). Enter COMPANY_ID of Study.
- **PI_STUDY_ID** (Mandatory). Enter OBJ_ID of Study.
- **PI_LIFECYCLE** (Mandatory). Enter Lifecycle context values like $LIFECYCLE$DEV, $LIFECYCLE$QC or $LIFECYCLE$PROD.

### 28.2 Create or Modify an Expression

In Oracle DMW, expressions in both validation checks and transformations are defined in the EXPR_OBJ_TYPE attribute of the DME_MAP_ENTITY_OBJ_TYPE. The EXPR_OBJ_TYPE attribute is of the DME_XFORM_EXPR_OBJ_TYPE object and its required values are:

- **EXPRESSION_ID** - If you are modifying an expression, enter the expression object ID.
- **EXPRESSION_MODE** - Enter one:
  - $EXPRMODE$CRITERIA if it is a criteria expression.
  - $EXPRMODE$EXPR if it is a column expression.
- **EXPRESSION_TEXT** - Enter the string representation of expression, adhering to the following rules:
  - All the values should be single quoted. Numeric values can be applied without single quotes.
The column names should be specified in the following format:
{ModelName.TableName.ColumnName}. For example, to specify a condition on
the 'AGE' column from 'DM' table in 'Source' data model that AGE should be
greater than 30, the expression_text should be passed as: {Source.DM.AGE} >
30

To specify a user-defined functions, the package should be available in the
view DME_PUB_XFM_EXPR_STATIC_PKGS_V. Refer to the static reference
details for the package that contains the function.

**EXPRESSION_ITEM_COLL** - Use this parameter only if you are calling a
function in the expression. This is a parameter of collection type DME_XFORM_ 
EXPR_ITEM_OBJ_COLL which is a table of DME_XFORM_EXPR_ITEM_OBJ_ 
TYPE object type.

For each expression_text, the collection must be populated with one record with
the following attribute assignments:

PARENT_ROW_ID = -1; ROW_ID=0; ROW_POS=1; ITEM_TYPE_RC=' $EXPITEMTYPE$GROUP'

If user-defined functions are used in expression_text, the collection should be
populated with a separate record for each distinct user-defined function used in
expression. The following attributes should be assigned for a record used for the
user defined function:

- **PARENT_ROW_ID** = 0;
- **ROW_POS** = 1;

---

**Note:** The above two values should always be 0 and 1, respectively.

- **ROW_ID** - The row_id must be unique and sequential starting from 1 for each
distinct user defined function used in the given expression.

- **ITEM_TYPE_RC** = ' $EXPITEMTYPE$FUNCTION';

Enter values for the following two attributes from the DME_XFM_EXPR_STATIC_ 
PKGS_V view.

- **ITEM** - Enter the name of the user-defined function.
- **STATIC_REFS** - Enter the function ID corresponding to the function used in
expression.

### 28.3 Using APIs to Create Custom Programs

In Oracle DMW you may need to write custom programs to handle special cases in
either transformations or validation checks. These programs must exist as user-defined
program objects in Oracle LSH. You can create them in the Oracle LSH user interface,
uploading the source code, or you can use APIs.

For more information, see the sections on custom programs in the *Oracle Health
Sciences Data Management Workbench User’s Guide* in the chapters on transformations
and validation checks. In addition, see the chapter on programs in the *Oracle Life
Sciences Data Hub Application Developer’s Guide*. 
This section includes the following topics:

- Section 29.1, "Flag-Related APIs"
- Section 29.2, "Flag Name-Related APIs"
- Section 29.3, "Model Categories-Related APIs"
- Section 29.4, "Action-Related APIs"

### 29.1 Flag-Related APIs

This section contains the following procedures related to flags:

- Section 29.1.1, "Set Flag"
- Section 29.1.2, "Get Flag"
- Section 29.1.3, "Get Flags on Data"
- Section 29.1.4, "Delete Flag"

#### 29.1.1 Set Flag

Use this API to set the state of a single flag for a given record.

**Name**  DME_PUB_FLAG_DATA.SetFlag

**Signature:**

```sql
PROCEDURE SETFLAG,
  ( P_API_VERSION IN VARCHAR2,
  P_INIT_MSG_LIST IN VARCHAR2 DEFAULT CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_COMMIT IN VARCHAR2 DEFAULT CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_VALIDATION_LEVEL IN NUMBER DEFAULT CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
  X_RETURN_STATUS OUT NOCOPY VARCHAR2,
  X_MSG_COUNT OUT NOCOPY NUMBER,
  X_MSG_DATA OUT NOCOPY VARCHAR2,
  PI_COMPANY_ID IN NUMBER,
  PI_TAB_OBJ_ID IN NUMBER,
  PI_SKEY_VALUE IN VARCHAR2,
  PI_FLAG_ID IN NUMBER,
  PI_FLAG_STATE IN VARCHAR2,
  );
```

**Parameters:** This API has standard parameters (see “Standard Parameters” on page 2-5) as well as the following parameters:
29.1.2 Get Flag

Use this API to retrieve information about a DMW flag setting.

**Name**  DME_PUB_FLAG_DATA.GetFlag

**Signature:**

```sql
PROCEDURE GETFLAG
(P_API_VERSION IN VARCHAR2,
P_INIT_MSG_LIST IN VARCHAR2 DEFAULT CDR_PUB_DEF_CONSTANTS.G_FALSE,
P_COMMIT IN VARCHAR2 DEFAULT CDR_PUB_DEF_CONSTANTS.G_FALSE,
P_VALIDATION_LEVEL IN NUMBER DEFAULT CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
X_RETURN_STATUS OUT NOCOPY VARCHAR2,
X_MSG_COUNT OUT NOCOPY NUMBER,
X_MSG_DATA OUT NOCOPY VARCHAR2,
PI_COMPANY_ID IN NUMBER,
PI_TAB_OBJ_ID IN NUMBER,
PI_SKEY_VALUE IN VARCHAR2,
PI_FLAG_ID IN NUMBER,
PO_FLAG_STATE OUT NOCOPY VARCHAR2);
```

**Parameters:**  This API has standard parameters (see "Standard Parameters" on page 2-5) as well as the following parameters:

- **PI_COMPANY_ID** (Mandatory). Enter the company ID.
- **PI_TAB_OBJ_ID** (Mandatory). Enter the obj_id of the table instance that contains the data record.
- **PI_SKEY_VALUE** (Mandatory). The surrogate key of the record.
- **PI_FLAG_ID** (Mandatory). The unique ID of the flag (from dme_flag_names)
- **PO_FLAG_STATE** (Mandatory). This output parameter inherits the flag state for this record.

29.1.3 Get Flags on Data

Use this API to retrieve information about all the flags and their values, assigned to a particular data record.

**Name**  DME_PUB_FLAG_DATA.GetFlagsOnData

**Signature:**

```sql
PROCEDURE GETFLAGSONDATA
(P_API_VERSION IN VARCHAR2,

```
Flag-Related APIs

Parameters: This API has standard parameters (see "Standard Parameters" on page 2-5) as well as the following parameters:

- **PI_COMPANY_ID** (Mandatory). Enter the company ID.
- **PI_TAB_OBJ_ID** (Mandatory). This input parameter is an object ID for a namings object of type $OBJTYPES$TABLEREF (table instance).
- **PI_SKEY_VALUE** (Mandatory). Enter the surrogate key of the record whose flags you want to retrieve.
- **PI.Include_Nulls** (Mandatory). Set this input parameter to 1 to include information for flags with NULL state, else 0.
- **PO_FLAGS** (Mandatory). This is a parameter of type CDR_NAME_VALUE_PAIR_COLL for the returned values (names and states).

29.1.4 Delete Flag

Use this API to delete an existing DMW Flag state.

Name: DME_PUB_FLAG_DATA.DeleteFlag

Signature:

PROCEDURE DELETEFLAG
(P_API_VERSION IN VARCHAR2,
P_INIT_MSG_LIST IN VARCHAR2 DEFAULT CDR_PUB_DEF_CONSTANTS.G_FALSE,
P_COMMIT IN VARCHAR2 DEFAULT CDR_PUB_DEF_CONSTANTS.G_FALSE,
P_VALIDATION_LEVEL IN NUMBER DEFAULT CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
X_RETURN_STATUS OUT NOCOPY VARCHAR2,
X_MSG_COUNT OUT NOCOPY NUMBER,
X_MSG_DATA OUT NOCOPY VARCHAR2,
PI_COMPANY_ID IN NUMBER,
PI_TAB_OBJ_ID IN NUMBER,
PI_SKEY_VALUE IN VARCHAR2,
PI_FLAG_ID IN NUMBER,
PO_FLAGS OUT NOCOPY CDR_NAME_VALUE_PAIR_COLL
);

Parameters: This API has standard parameters (see "Standard Parameters" on page 2-5) as well as the following parameters:

- **PI_COMPANY_ID** (Mandatory). Enter the company ID.
- **PI_TAB_OBJ_ID** (Mandatory). This input parameter is an object id for a namings object of type $OBJTYPES$TABLEREF (table instance).
- **PI_SKEY_VALUE** (Mandatory). The surrogate key of the record.
Flag Name-Related APIs

■ PI_FLAG_ID (Mandatory). This input parameter is the ID of the flag whose state you want to remove.

29.2 Flag Name-Related APIs

Use the following APIs to retrieve DMW flag names definitions and states. Use the first version of these APIs if you know the flag ID; use the second one if you know its name.

This section includes the following:

■ Section 29.2.1, "Get Flag Name Definition, Version 1"
■ Section 29.2.2, "Get Flag Name Definition, Version 2"
■ Section 29.2.3, "Get Flag Name Definitions"
■ Section 29.2.4, "Get Flag States"

29.2.1 Get Flag Name Definition, Version 1

Use this API to retrieve information about a DMW flag name definition. Use this API if you know the Flag Name ID.

Name DME_PUB_FLAG_NAME.GetFlagName

Signature:

PROCEDURE GETFLAGNAME
(P_API_VERSION IN VARCHAR2,
P_INIT_MSG_LIST IN VARCHAR2 DEFAULT CDR_PUB_DEF_CONSTANTS.G_FALSE,
P_COMMIT IN VARCHAR2 DEFAULT CDR_PUB_DEF_CONSTANTS.G_FALSE,
P_VALIDATION_LEVEL IN NUMBER DEFAULT CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
X_RETURN_STATUS OUT NOCOPY VARCHAR2,
X_MSG_COUNT OUT NOCOPY NUMBER,
X_MSG_DATA OUT NOCOPY VARCHAR2,
PI_COMPANY_ID IN NUMBER,
PI_FLAG_NAME_ID IN NUMBER,
PIO_DME_FLAG_NAME IN OUT NOCOPY DME_FLAG_NAME_TYPE,
);

Parameters: This API has standard parameters (see "Standard Parameters" on page 2-5) as well as the following parameters:

■ PI_COMPANY_ID (Mandatory). Enter the company ID.
■ PI_FLAG_NAME_ID (Mandatory). Enter the ID of the flag name definition to be retrieved.
■ PIO_DME_FLAG_NAME (Mandatory). This is a parameter of type DME_FLAG_DEF_NAME for the returned values.

29.2.2 Get Flag Name Definition, Version 2

Use this API to retrieve information about a DMW flag name definition. Use this API if you know the Flag Name.

Name DME_PUB_FLAG_NAME.GetFlagName

Signature:
PROCEDURE GETFLAGNAME
(P_API_VERSION IN VARCHAR2,
P_INIT_MSG_LIST IN VARCHAR2 DEFAULT CDR_PUB_DEF_CONSTANTS.G_FALSE,
P_COMMIT IN VARCHAR2 DEFAULT CDR_PUB_DEF_CONSTANTS.G_FALSE,
P_VALIDATION_LEVEL IN NUMBER DEFAULT CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
X_RETURN_STATUS OUT NOCOPY VARCHAR2,
X_MSG_COUNT OUT NOCOPY NUMBER,
X_MSG_DATA OUT NOCOPY VARCHAR2,
PI_COMPANY_ID IN NUMBER,
PI_FLAG_NAMESTR IN VARCHAR2,
PIO_DME_FLAG_NAME IN OUT NOCOPY DME_FLAG_NAME_TYPE,
);

Parameters:  This API has standard parameters (see "Standard Parameters" on page 2-5) as well as the following parameters:

- **PI_COMPANY_ID** (Mandatory). Enter the company ID.
- **PI_FLAG_NAMESTR** (Mandatory). Enter the name of the flag name definition to be retrieved.
- **PIO_DME_FLAG_NAME** (Mandatory). This is a parameter of type DME_FLAG_DEF_NAME for the returned values.

29.2.3 Get Flag Name Definitions

Use this API to retrieve information about DMW flag name definitions.

Name  DME_PUB_FLAG_NAME.GetFlagNames

Signature:

PROCEDURE GETFLAGNAMES
( P_API_VERSION IN VARCHAR2,
P_INIT_MSG_LIST IN VARCHAR2 DEFAULT CDR_PUB_DEF_CONSTANTS.G_FALSE,
P_COMMIT IN VARCHAR2 DEFAULT CDR_PUB_DEF_CONSTANTS.G_FALSE,
P_VALIDATION_LEVEL IN NUMBER DEFAULT CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
X_RETURN_STATUS OUT NOCOPY VARCHAR2,
X_MSG_COUNT OUT NOCOPY NUMBER,
X_MSG_DATA OUT NOCOPY VARCHAR2,
PI_CATEGORY_ID IN NUMBER DEFAULT NULL,
PI_SUBJECT_VISIT IN VARCHAR2 DEFAULT NULL,
PI_USER_SETTABLE IN VARCHAR2 DEFAULT NULL,
PO_FLAG_NAMES OUT NOCOPY DME_FLAG_NAME_COLL,
);

Parameters:  This API has standard parameters (see "Standard Parameters" on page 2-5) as well as the following parameters:

- **PI_CATEGORY_ID** (Optional). This input parameter is set to null to retrieve flags in any category. To get matching flags from a given category, you must provide a valid category ID.
- **PI_SUBJECT_VISIT** (Optional). This input parameter is set to null to match flags irrespective of their being marked as subject_visit flags or not. To retrieve matching subject_visit flags, enter either $YESNO$YES or $YESNO$NO as appropriate.
- **PI_USER_SETTABLE** (Optional). This input parameter is set to null to match flags whether or not they are settable by the user. To match user-settable options, provide either $YESNO$YES or $YESNO$NO as appropriate.
29.2.4 Get Flag States

Use this API to retrieve information about all the DMW flag states for a single flag.

**Name**  DME_PUB_FLAG_STATE.GetFlagStates

**Signature:**

```sql
PROCEDURE GETFLAGSTATES
(P_API_VERSION IN VARCHAR2,
P_INIT_MSG_LIST IN VARCHAR2 DEFAULT CDR_PUB_DEF_CONSTANTS.G_FALSE,
P_COMMIT IN VARCHAR2 DEFAULT CDR_PUB_DEF_CONSTANTS.G_FALSE,
P_VALIDATION_LEVEL IN NUMBER DEFAULT CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
X_RETURN_STATUS OUT NOCOPY VARCHAR2,
X_MSG_COUNT OUT NOCOPY NUMBER,
X_MSG_DATA OUT NOCOPY VARCHAR2,
PI_COMPANY_ID IN NUMBER,
PI_FLAG_NAME_ID IN NUMBER,
PO_DME_FLAG_STATES OUT NOCOPY DME_FLAG_STATE_COLL,
);
```

**Parameters:** This API has standard parameters (see "Standard Parameters" on page 2-5) as well as the following parameters:

- **PI_COMPANY_ID** (Mandatory). Enter the company ID.
- **PI_FLAG_NAME_ID** (Mandatory). This input parameter is the ID of the flag name definition that contains the flag value.
- **PO_DME_FLAG_STATES** (Mandatory). This is a parameter of type DME_FLAG_STATE_COLL for the returned information.

29.3 Model Categories-Related APIs

This section includes the following:

- Section 29.3.1, "Create Model Flag Category Mapping"
- Section 29.3.2, "Get Categories for Model"

29.3.1 Create Model Flag Category Mapping

Use this API to create a map entry between a model type and subtype and a flag category.

**Name**  DME_PUB_MODEL_FLAGCAT.CreateModelFlagcatMap

**Signature:**

```sql
PROCEDURE CREATEMODELFLAGCATMAP,
(P_API_VERSION IN VARCHAR2,
P_INIT_MSG_LIST IN VARCHAR2 DEFAULT CDR_PUB_DEF_CONSTANTS.G_FALSE,
P_COMMIT IN VARCHAR2 DEFAULT CDR_PUB_DEF_CONSTANTS.G_FALSE,
P_VALIDATION_LEVEL IN NUMBER DEFAULT CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
X_RETURN_STATUS OUT NOCOPY VARCHAR2,
X_MSG_COUNT OUT NOCOPY NUMBER,
X_MSG_DATA OUT NOCOPY VARCHAR2,
```
PI_COMPANY_ID IN NUMBER,
PI_MODEL_TYPE_RC IN VARCHAR2,
PI_MODEL_SUBTYPE_RC IN VARCHAR2,
PI_CATEGORY_ID IN NUMBER,
};

**Parameters:** This API has standard parameters (see "Standard Parameters" on page 2-5) as well as the following parameters:

- **PI_COMPANY_ID** (Mandatory). Enter the company ID.
- **PI_MODEL_TYPE_RC** (Mandatory). Enter a model type from the lookup CDR_MODELTYPES such as $MODELTYPE$INPUT
  
  If you want to map a flag category to input models that come from files, pass these two arguments as '$MODELTYPE$INPUT' and '$INPUTMODTYPE$FILE'.
  
  If you want to map a flag category to input models from InForm, pass these two arguments as '$MODELTYPE$INPUT' and '$INPUTMODTYPE$INFORM'.
  
  If you want to map a flag category to target models, pass these arguments as '$MODELTYPE$TARGET' and NULL.
- **PI_MODEL_SUBTYPE_RC** (Mandatory). Enter a model subtype; for example, $MODELTYPE$INPUT or $MODELTYPE$TARGET
- **PI_CATEGORY_ID** (Mandatory). Enter the unique ID of the category (from dme_categories in the Administration Area of the DMW User Interface).

### 29.3.2 Get Categories for Model

Use this API to get all categories that are valid for a model type and subtype.

**Name** DME_PUB_MODEL_FLAGCAT.GetCategoriesForModel

**Signature:**

```
PROCEDURE GETCATEGORYSFORMODEL,
P_API_VERSION IN VARCHAR2,
P_INIT_MSG_LIST IN VARCHAR2 DEFAULT CDR_PUB_DEF_CONSTANTS.G_FALSE,
P_COMMIT IN VARCHAR2 DEFAULT CDR_PUB_DEF_CONSTANTS.G_FALSE,
P_VALIDATION_LEVEL IN NUMBER DEFAULT CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
X_RETURN_STATUS OUT NOCOPY VARCHAR2,
X_MSG_COUNT OUT NOCOPY NUMBER,
X_MSG_DATA OUT NOCOPY VARCHAR2,
PI_COMPANY_ID IN NUMBER,
PI_MODEL_TYPE_RC IN VARCHAR2,
PI_MODEL_SUBTYPE_RC IN VARCHAR2,
PO_CATEGORIES OUT NOCOPY DME_NUMBER_COLL,
};
```

**Parameters:** This API has standard parameters (see "Standard Parameters" on page 2-5) as well as the following parameters:

- **PI_COMPANY_ID** (Mandatory). Enter the company ID.
- **PI_MODEL_TYPE_RC** (Mandatory). Enter a model type such as $MODELTYPE$INPUT, from the lookup CDR_MODELTYPES.
- **PI_MODEL_SUBTYPE_RC** (Mandatory). Enter a model subtype; for example, $INPUTMODTYPE$FILE.
- **PO_CATEGORIES** Output parameter returns an array of category IDs.
29.4 Action-Related APIs

Use the following APIs to perform tasks related to discrepancy actions:

- Section 29.4.1, "Create Discrepancy Action"
- Section 29.4.2, "Get Disc Action, Version 1"
- Section 29.4.3, "Get Disc Action, Version 2"
- Section 29.4.4, "Update Discrepancy Action"
- Section 29.4.5, "Delete Discrepancy Action"

29.4.1 Create Discrepancy Action

Use this API to create a new discrepancy action.

Name  DME_PUB_DISC_ACTION.CreateDiscAction

Signature:

PROCEDURE CREATE_DISC_ACTION
(P_API_VERSION IN VARCHAR2,
 P_INIT_MSG_LIST IN VARCHAR2 DEFAULT CDR_PUB_DEF_CONSTANTS.G_FALSE,
 P_COMMIT IN VARCHAR2 DEFAULT CDR_PUB_DEF_CONSTANTS.G_FALSE,
 P_VALIDATION_LEVEL IN NUMBER DEFAULT CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
 X_RETURN_STATUS OUT NOCOPY VARCHAR2,
 X_MSG_COUNT OUT NOCOPY NUMBER,
 X_MSG_DATA OUT NOCOPY VARCHAR2,
 PIO_DME_DISC_ACTION IN OUT NOCOPY DME_DISC_ACTION_TYPE,
) ;

Parameters:  This API has standard parameters (see "Standard Parameters" on page 2-5) as well as the following parameters:

- PIO_DME_DISC_ACTION  (Mandatory). This is a parameter of type DME_DISC_ACTION_TYPE that contains action attributes.
  Enter the Start State, one of Candidate, Open, Answered, Cancelled, Closed. (In the API those are specified as $DISC_STATE$CANDIDATE, $DISC_STATE$OPEN, $DISC_STATE$ANSWERED, $DISC_STATE$CANCELLED, and $DISC_STATE$CLOSED).

29.4.2 Get Disc Action, Version 1

Use this API to retrieve information about a DMW Discrepancy Action.

Name  DME_PUB_DISC_ACTION.GetDiscAction

Signature:

PROCEDURE GET_DISC_ACTION
(P_API_VERSION IN VARCHAR2,
 P_INIT_MSG_LIST IN VARCHAR2 DEFAULT CDR_PUB_DEF_CONSTANTS.G_FALSE,
 P_COMMIT IN VARCHAR2 DEFAULT CDR_PUB_DEF_CONSTANTS.G_FALSE,
 P_VALIDATION_LEVEL IN NUMBER DEFAULT CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
 X_RETURN_STATUS OUT NOCOPY VARCHAR2,
 X_MSG_COUNT OUT NOCOPY NUMBER,
 X_MSG_DATA OUT NOCOPY VARCHAR2,
 PI_ACTION_ID IN NUMBER,
 PIO_DME_DISC_ACTION IN OUT NOCOPY DME_DISC_ACTION_TYPE,
29.4.3 Get Disc Action, Version 2

Use this API to retrieve information about a DMW Discrepancy Action.

**Name** DME_PUB_DISC_ACTION.GetDiscAction

**Signature:**

```sql
PROCEDURE GETDISCACTION,
( P_API_VERSION IN VARCHAR2,
  P_INIT_MSG_LIST IN VARCHAR2 DEFAULT CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_COMMIT IN VARCHAR2 DEFAULT CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_VALIDATION_LEVEL IN NUMBER DEFAULT CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
  X_RETURN_STATUS OUT NOCOPY VARCHAR2,
  X_MSG_COUNT OUT NOCOPY NUMBER,
  X_MSG_DATA OUT NOCOPY VARCHAR2,
  PI_ACTION_NAME IN VARCHAR2,
  PIO_DME_DISC_ACTION IN OUT NOCOPY DME_DISC_ACTION_TYPE,
); 
```

**Parameters:** This API has standard parameters (see “Standard Parameters” on page 2-5) as well as the following parameters:

- **PI_ACTION_NAME** (Mandatory). Provide the name of the discrepancy action to retrieve information for.
- **PIO_DME_DISC_ACTION** (Mandatory). This is a parameter of type DME_DISC_ACTION_TYPE for the returned values.

29.4.4 Update Discrepancy Action

Use this API to modify an existing DMW Discrepancy Action.

**Name** DME_PUB_DISC_ACTION.UpdateDiscAction

**Signature:**

```sql
PROCEDURE UPDATEDISCACTION,
( P_API_VERSION IN VARCHAR2,
  P_INIT_MSG_LIST IN VARCHAR2 DEFAULT CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_COMMIT IN VARCHAR2 DEFAULT CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_VALIDATION_LEVEL IN NUMBER DEFAULT CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
  X_RETURN_STATUS OUT NOCOPY VARCHAR2,
  X_MSG_COUNT OUT NOCOPY NUMBER,
  X_MSG_DATA OUT NOCOPY VARCHAR2,
  PIO_DME_DISC_ACTION IN OUT NOCOPY DME_DISC_ACTION_TYPE,
); 
```
Parameters: This API has standard parameters (see "Standard Parameters" on page 2-5) as well as the following parameter:

- **PIO_DME_DISC_ACTION** (Mandatory). This is a parameter of type DME_DISC_ACTION_TYPE that contains the discrepancy action attributes to be updated.

### 29.4.5 Delete Discrepancy Action

Use this API to delete an existing DMW Discrepancy Action.

**Name** DME_PUB_DISC_ACTION.DeleteDiscAction

**Signature:**

```
PROCEDURE DELETEDISCACTION,
(P_API_VERSION IN VARCHAR2,
P_INIT_MSG_LIST IN VARCHAR2 DEFAULT CDR_PUB_DEF_CONSTANTS.G_FALSE,
P_COMMIT IN VARCHAR2 DEFAULT CDR_PUB_DEF_CONSTANTS.G_FALSE,
P_VALIDATION_LEVEL IN NUMBER DEFAULT CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
X_RETURN_STATUS OUT NOCOPY VARCHAR2,
X_MSG_COUNT OUT NOCOPY NUMBER,
X_MSG_DATA OUT NOCOPY VARCHAR2,
PI_DME_DISC_ACTION_ID IN NUMBER,
);
```

Parameters: This API has standard parameters (see "Standard Parameters" on page 2-5) as well as the following parameters:

- **PI_DME_DISC_ACTION_ID** (Mandatory). This is a parameter of type NUMBER, that contains the unique ID of the category to be disabled.
This is a public interface for all operations related to Code Lists, including creation, deletion, modification, and checking in and out of these objects.

This section includes the Code List APIs from the following package: DME_PUB_CODE_LISTS.

30.1 Create and Modify Code Lists

This section contains the following public APIs:

- Section 30.1.1, "Create a Code List"
- Section 30.1.2, "Modify a Code List"
- Section 30.1.3, "Remove a Code List"
- Section 30.1.4, "Check In a Code List"
- Section 30.1.5, "Check Out a Code List"
- Section 30.1.6, "Add Values to a Code List"
- Section 30.1.7, "Remove Values from a Code List"
- Section 30.1.8, "Get Code List Details for a Given Column"

30.1.1 Create a Code List

Use this API to create a code list under a library domain. Code lists created with this API can be verified through public view 'DME_PUB_CODELIST_V'. For values refer 'DME_PUB_CODELIST_VALUES_V'.

**Name**  DME_PUB_CODE_LISTS.CreateCodeList

**Signature:**

```sql
PROCEDURE CREATECODELIST
(P_API_VERSION IN NUMBER,
P_INIT_MSG_LIST IN VARCHAR2 DEFAULT CDR_PUB_DEF_CONSTANTS.G_FALSE,
P_COMMIT IN VARCHAR2 DEFAULT CDR_PUB_DEF_CONSTANTS.G_FALSE,
P_VALIDATION_LEVEL IN NUMBER DEFAULT CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
P_NAMING IN OUT NOCOPY cdr_naming_version_obj_type,
P_CODELISTCOLL IN CDR_CODE_LIST_COLL,
X_RETURN_STATUS OUT NOCOPY VARCHAR2,
X_MSG_COUNT OUT NOCOPY NUMBER,
X_MSG_DATA OUT NOCOPY VARCHAR2,
);
```
**Parameters:** This API has standard parameters (see "Standard Parameters" on page 2-5) as well as the following parameters:

- **P_NAMING** (Mandatory). This is a parameter of type CDR_NAMING_VERSION_OBJ_TYPE that contains naming attributes. The attributes required for this API are:
  - COMPANY_ID
  - NAME
  - NAMESPACE_OBJ_ID
  - NAMESPACE_OBJ_VER
  - NAMESPACE_START_OBJ_VER as 1
  - NAMESPACE_END_OBJ_VER as 999999
  - DESCRIPTION
  - OBJECT_TYPE_RC as '$OBJTYPES$CODELIST'

- **P_CODELISTCOLL** (Optional). This is a parameter of type CDR_CODE_LIST_COLL. Collection of code value pairs. For example:

  CDR_CODE_LIST_COLL(CDR_CODE_LIST_OBJ_TYPE('CDE1', 'VAL1'), CDR_CODE_LIST_OBJ_TYPE('CDE2', 'VAL2');

### 30.1.2 Modify a Code List

Use this API to modify the naming attributes of a code list. Code lists modified with this API can be verified through public view 'DME_PUB_CODELIST_V'.

**Name** DME_PUB_CODE_LISTS.ModifyCodeList

**Signature:**

```sql
PROCEDURE MODIFYCODELIST
  (P_API_VERSION IN NUMBER,
   P_INIT_MSG_LIST IN VARCHAR2 DEFAULT CDR_PUB_DEF_CONSTANTS.G_FALSE,
   P_COMMIT IN VARCHAR2 DEFAULT CDR_PUB_DEF_CONSTANTS.G_FALSE,
   P_VALIDATION_LEVEL IN NUMBER DEFAULT CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
   P_NAMING IN OUT NOCOPY CDR_NAMING_VERSION_OBJ_TYPE,
   X_RETURN_STATUS OUT NOCOPY VARCHAR2,
   X_MSG_COUNT OUT NOCOPY NUMBER,
   X_MSG_DATA OUT NOCOPY VARCHAR2,
 );
```

**Parameters:** This API has standard parameters (see "Standard Parameters" on page 2-5) as well as the following parameters:

- **P_NAMING** (Mandatory). This is a parameter of type CDR_NAMING_VERSION_OBJ_TYPE that contains naming attributes. Naming details of the Code list to which the code-value pairs have to be modified. Whole naming object needs to be populated for the object which needs to be modified. API 'CDR_PUB_DEF_FACTORY_SUPPORT.GETNAMINGOBJECT' can be used to populate naming object, and then the values, which user wanted to be modified, e.g: NAME, DESCRIPTION.
30.1.3 Remove a Code List

Use this API to remove a code list. After this API is executed, public view 'DME_PUB_CODELIST_V' will not display the code list.

**Name**  DME_PUB_CODE_LISTS.RemoveCodeList

**Signature:**

```sql
PROCEDURE REMOVECODELIST
(P_API_VERSION IN NUMBER,
 P_INIT_MSG_LIST IN VARCHAR2 DEFAULT CDR_PUB_DEF_CONSTANTS.G_FALSE,
 P_COMMIT IN VARCHAR2 DEFAULT CDR_PUB_DEF_CONSTANTS.G_FALSE,
 P_VALIDATION_LEVEL IN NUMBER DEFAULT CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
 P_NAMING IN OUT NOCOPY cdr_naming_version_obj_type,
 X_RETURN_STATUS OUT NOCOPY VARCHAR2,
 X_MSG_COUNT OUT NOCOPY NUMBER,
 X_MSG_DATA OUT NOCOPY VARCHAR2);```

**Parameters:**  This API has standard parameters (see "Standard Parameters" on page 2-5) as well as the following parameters:

- **P_NAMING** (Mandatory). This is a parameter of type CDR_NAMING_VERSION_OBJ_TYPE that contains naming attributes. Naming details of the code list to which the code-value pairs have to be removed. Whole naming object needs to be populated for the object which needs to be removed. API 'CDR_PUB_DEF_FACTORY_SUPPORT.GETNAMINGOBJECT' can be used to populate naming object.

30.1.4 Check In a Code List

Use this API to check in a code list. Changed status can be verified through public view 'DME_PUB_CODELIST_V'.

**Name**  DME_PUB_CODE_LISTS.CheckinCodeList

**Signature:**

```sql
PROCEDURE CHECKINCODELIST
(P_API_VERSION IN NUMBER,
 P_INIT_MSG_LIST IN VARCHAR2 DEFAULT CDR_PUB_DEF_CONSTANTS.G_FALSE,
 p_commit IN VARCHAR2 DEFAULT CDR_PUB_DEF_CONSTANTS.G_FALSE,
 p_validation_level IN NUMBER DEFAULT CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
 p_naming IN OUT NOCOPY cdr_naming_version_obj_type,
 x_return_status OUT NOCOPY VARCHAR2,
 x_msg_count OUT NOCOPY NUMBER,
 x_msg_data OUT NOCOPY VARCHAR2);```

**Parameters:**  This API has standard parameters (see "Standard Parameters" on page 2-5) as well as the following parameters:

- **P_NAMING** (Mandatory). This is a parameter of type CDR_NAMING_VERSION_OBJ_TYPE that contains naming attributes. Whole naming object needs to be populated for the object which needs to be checked-in. API 'CDR_PUB_DEF_FACTORY_SUPPORT.GETNAMINGOBJECT' can be used to populate naming object.
30.1.5 Check Out a Code List

Use this API to check out a code list. Changed status can be verified through public view 'DME_PUB_CODELIST_V'.

**Name**  DME_PUB_CODELISTS.CheckoutCodeList

**Signature:**

```
PROCEDURE CHECKOUTCODELIST
( P_API_VERSION      IN NUMBER ,
  P_INIT_MSG_LIST    IN VARCHAR2 DEFAULT CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_COMMIT           IN VARCHAR2 DEFAULT CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_VALIDATION_LEVEL IN NUMBER DEFAULT CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
  P_NAMING           IN OUT NOCOPY cdr_naming_version_obj_type,
  X_RETURN_STATUS OUT NOCOPY VARCHAR2,
  X_MSG_COUNT OUT NOCOPY     NUMBER,
  X_MSG_DATA OUT NOCOPY      VARCHAR2)
);```

**Parameters:**  This API has standard parameters (see "Standard Parameters" on page 2-5) as well as the following parameters:

- **P_NAMING** (Mandatory). This is a parameter of type CDR_NAMING_VERSION_OBJ_TYPE that contains naming attributes. Whole naming object needs to be populated for the object which needs to be checked-out. API 'CDR_PUB_DEF_FACTORY_SUPPORT.GETNAMINGOBJECT' can be used to populate naming object.

30.1.6 Add Values to a Code List

Use this API to add code value pairs to a code list. Code list values created with this API can be verified through public view 'DME_PUB_CODELIST_VALUES_V'.

**Name**  DME_PUB_CODELISTS.AddCodeListValues

**Signature:**

```
PROCEDURE ADDCODELISTVALUES
( P_API_VERSION      IN NUMBER ,
  P_INIT_MSG_LIST    IN VARCHAR2 DEFAULT CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_COMMIT           IN VARCHAR2 DEFAULT CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_VALIDATION_LEVEL IN NUMBER DEFAULT CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
  P_NAMING           IN OUT NOCOPY cdr_naming_version_obj_type ,
  P_CODELISTCOLL     IN CDR_CODE_LIST_COLL ,
  X_RETURN_STATUS OUT NOCOPY VARCHAR2 ,
  X_MSG_COUNT OUT NOCOPY     NUMBER ,
  X_MSG_DATA OUT NOCOPY      VARCHAR2)
);```

**Parameters:**  This API has standard parameters (see "Standard Parameters" on page 2-5) as well as the following parameters:

- **P_NAMING** (Mandatory). This is a parameter of type CDR_NAMING_VERSION_OBJ_TYPE that contains naming attributes. Whole naming object needs to be populated for the object which needs code values to be added to it. API 'CDR_PUB_DEF_FACTORY_SUPPORT.GETNAMINGOBJECT' can be used to populate naming object.
30.1.7 Remove Values from a Code List

Use this API to remove code value pairs to a code list. After this API is executed, public view ‘DME_PUB_CODELIST_V’ will not display the code list values.

Name  DME_PUB_CODELISTS.AddCodeListValues

Signature:

PROCEDURE REMOVECODELISTVALUES
(P_API_VERSION      IN NUMBER,
P_INIT_MSG_LIST    IN VARCHAR2 DEFAULT CDR_PUB_DEF_CONSTANTS.G_FALSE,
P_COMMIT           IN VARCHAR2 DEFAULT CDR_PUB_DEF_CONSTANTS.G_FALSE,
P_VALIDATION_LEVEL IN NUMBER DEFAULT CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
P_NAMING           IN OUT NOCOPY cdr_naming_version_obj_type,
P_CODELISTCOLL     IN CDR_CODE_LIST_COLL,
X_RETURN_STATUS OUT NOCOPY VARCHAR2,
X_MSG_COUNT OUT NOCOPY     NUMBER,
X_MSG_DATA OUT NOCOPY      VARCHAR2
);

Parameters:  This API has standard parameters (see “Standard Parameters” on page 2-5) as well as the following parameters:

■ P_NAMING (Mandatory). This is a parameter of type CDR_NAMING_VERSION_OBJ_TYPE that contains naming attributes. Whole naming object needs to be populated for the object which needs code values to be added to it. API ‘CDR_PUB_DEF_FACTORY_SUPPORT.GETNAMINGOBJECT’ can be used to populate naming object.

■ P_CODELISTCOLL (Mandatory). This is a parameter of type CDR_CODE_LIST_COLL. Collection of code value pairs. For example:

CDR_CODE_LIST_COLL(CDR_CODE_LIST_OBJ_TYPE('CDE1', 'VAL1'), CDR_CODE_LIST_OBJ_TYPE('CDE2', 'VAL2');

30.1.8 Get Code List Details for a Given Column

Use this API to get the code list details for a given column.

Name  DME_PUB_CODELISTS.getCodeListDetail

Signature:

PROCEDURE GETCODELISTDETAIL
(P_API_VERSION      IN NUMBER ,
P_INIT_MSG_LIST    IN VARCHAR2 DEFAULT CDR_PUB_DEF_CONSTANTS.G_FALSE,
P_COMMIT           IN VARCHAR2 DEFAULT CDR_PUB_DEF_CONSTANTS.G_FALSE,
P_VALIDATION_LEVEL IN NUMBER DEFAULT CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
X_RETURN_STATUS OUT NOCOPY VARCHAR2 ,
X_MSG_COUNT OUT NOCOPY     NUMBER,
X_MSG_DATA OUT NOCOPY      VARCHAR2,
PI_COLCOMPANYID    IN cdr_naming_versions.company_id%type,
PI_COLOBJID        IN cdr_naming_versions.obj_id%type,
PI_COLOBJVER       IN cdr_naming_versions.obj_ver%type,
Create and Modify Code Lists

```sql
PROCEDURE cdrRemoveColumns
PO_CODELISTID   OUT NOCOPY cdr_naming_versions.obj_id%type,
PO_CODELISTVER  OUT NOCOPY cdr_naming_versions.obj_Ver%type
); PARAMETERS: This API has standard parameters (see "Standard Parameters" on page 2-5) as well as the following parameters:
- **PI_COLCOMPANYID** (Mandatory). COMPANY_ID of a Column.
- **PI_COLOBJID** (Mandatory). Object Identifier of a Column
- **PI_COLOBJVER** (Mandatory). Object Version of a Column
- **PO_CODELISTID**. This is out parameter. It returns the Object Id of associated code list.
- **PO_CODELISTVER**. This is out parameter. It returns the Object Version of associated code list.
This is a public interface for all operations related to Validation Checks Batch(es), including creation, deletion, modification, and checking in and out of these objects. It also includes information on how to create or modify expressions.

This section includes the Validation Checks APIs from the following package: DME_PUB_VALIDATION_CHECK.

### 31.1 Create and Modify Validation Checks and Batches

This section contains the following public APIs:

- Section 31.1.1, "Create a Validation Check Batch"
- Section 31.1.2, "Modify a Validation Check Batch"
- Section 31.1.3, "Remove Validation Check Batch(es)"
- Section 31.1.4, "Create a Validation Check"
- Section 31.1.5, "Update a Validation Check"
- Section 31.1.6, "Install a Validation Check Batch"
- Section 31.1.7, "Submit a Validation Check Batch"
- Section 31.1.8, "Check In a Validation Check Batch"
- Section 31.1.9, "Check Out a Validation Check Batch"
- Section 31.1.10, "Undo Checkout For a Validation Check Batch"
- Section 31.1.11, "Update Validation Status of a Validation Check Batch"
- Section 31.1.12, "Upgrade a Validation Check Batch"
- Section 31.1.13, "Remove Validation Check(s)"
- Section 31.1.14, "Enable or Disable Validation Checks"
- Section 31.1.15, "Reorder Validation Checks"

#### 31.1.1 Create a Validation Check Batch

Use this API to create a validation check batch.

**Name**  
DME_PUB_VALIDATION_CHECK.CreateValidationCheckBatch

**Signature:**  
PROCEDURE CREATEVALIDATIONCHECKBATCH,
Create and Modify Validation Checks and Batches

Parameters:

This API has standard parameters (see "Standard Parameters" on page 2-5) as well as the following parameters:

- **PI_SOURCEMODELCOMPID** (Mandatory). Enter COMPANY_ID of Source Data Model.
- **PI_SOURCEMODELOBJID** (Mandatory). Enter the obj_id of the table instance which contains the data record.
- **PI_SOURCEMODELOBJVER** (Mandatory). Enter OBJ_VER of Source Data Model.
- **PI_BATCHNAME** (Mandatory). Enter Batch Name for new VC Batch.
- **PI_BATCHDESCRIPTION** (Optional). Enter Batch Description if you want for new VC Batch.
- **PI_ISORDEREDFOREXECUTION** (Optional). Enter $YESNO$YES, if Validation Checks under new Batch must be executed in particular order. Otherwise enter $YESNO$NO. If entered as NULL, systems defaults it to $YESNO$NO.
- **PI_CANBETRIGGERED** (Optional). Enter $YESNO$YES to allow the successful completion of a transformation writing to the clinical data model that this batch runs against to trigger the execution of this batch. Otherwise enter $YESNO$NO. If entered as NULL, systems defaults it to $YESNO$NO.

31.1.2 Modify a Validation Check Batch

Use this API to modify a validation check batch.

**Name**  
DME_PUB_VALIDATION_CHECK.UpdateValidationCheckBatch

**Signature:**

PROCEDURE UPDATEVALIDATIONCHECKBATCH
(  
P_API_VERSION IN VARCHAR2,
P_INIT_MSG_LIST IN VARCHAR2 DEFAULT CDR_PUB_DEF_CONSTANTS.G_FALSE,
P_COMMIT IN VARCHAR2 DEFAULT CDR_PUB_DEF_CONSTANTS.G_FALSE,
P_VALIDATION_LEVEL IN NUMBER default CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
X_RETURN_STATUS OUT NOCOPY VARCHAR2,
X_MSG_COUNT OUT NOCOPY NUMBER,
X_MSG_DATA OUT NOCOPY VARCHAR2,
PI_BATCHMAPCOMPID IN OUT NOCOPY CDR_NAMINGS.COMPANY_ID%TYPE,
PI_BATCHMAPOBJID IN OUT NOCOPY CDR_NAMINGS.OBJ_ID%TYPE,
PI_BATCHMAPOBJVER IN OUT NOCOPY CDR_NAMING_VERSIONS.OBJ_VER%TYPE,
PI_BATCHNAME IN OUT NOCOPY CDR_NAMINGS.NAME%TYPE,
);
PI_BATCHDESCRIPTION IN CDR_NAMING_VERSIONS.DESCRIPTION%TYPE,
PI_ISORDEREDFOREXECUTION IN VARCHAR2,
PI_CANBETRIGGERED IN VARCHAR DEFAULT '$YESNO$NO'
);

Parameters: This API has standard parameters (see "Standard Parameters" on page 2-5) as well as the following parameters:

- **PI_BATCHMAPCOMPID** (Mandatory). Enter COMPANY_ID of Validation Checks Batch Map.
- **PI_BATCHMAPOBJID** (Mandatory). Enter OBJ_ID of Validation Checks Batch Map.
- **PI_BATCHMAPOBJVER** (Mandatory). Enter OBJ_VER of Validation Checks Batch Map.
- **PI_BATCHNAME** (Mandatory). Enter Batch Name for new Validation Checks Batch.
- **PI_BATCHDESCRIPTION** (Optional). Enter Batch Description for new Validation Checks Batch.
- **PI_ISORDEREDFOREXECUTION** (Optional). Enter $YESNO$YES, if Validation Checks under new Batch must be executed in particular order. Otherwise enter $YESNO$NO. If entered as NULL, systems defaults it to $YESNO$NO.
- **PI_CANBETRIGGERED** (Optional). Enter $YESNO$YES to allow the successful completion of a transformation writing to the clinical data model that this batch runs against to trigger the execution of this batch. Otherwise enter $YESNO$NO. If entered as NULL, systems defaults it to $YESNO$NO.

### 31.1.3 Remove Validation Check Batch(es)

Use this API to remove Validation Check batch(es).

**Name**  DME_PUB_VALIDATION_CHECK.RemoveValidationCheckBatches

**Signature:**

PROCEDURE REMOVEVALIDATIONCHECKBATCHES
(P_API_VERSION IN VARCHAR2,
P_INIT_MSG_LIST IN VARCHAR2 DEFAULT CDR_PUB_DEF_CONSTANTS.G_FALSE,
P_COMMIT IN VARCHAR2 DEFAULT CDR_PUB_DEF_CONSTANTS.G_FALSE,
P_VALIDATION_LEVEL IN NUMBER DEFAULT CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
X_RETURN_STATUS OUT NOCOPY VARCHAR2,
X_MSG_COUNT OUT NOCOPY NUMBER,
X_MSG_DATA OUT NOCOPY VARCHAR2,
PIO_VC_BATCH_COLL IN OUT NOCOPY CDR_NAMING_LIST_COLL
);

Parameters: This API has standard parameters (see "Standard Parameters" on page 2-5) as well as the following parameters:

- **PIO_VC_BATCH_COLL** (Mandatory). This is a parameter of collection type CDR_NAMING_LIST_COLL that contains CDR Naming Version attributes. The following attributes are required for Validation Checks Batch maps:
  - COMPANY_ID
  - OBJ_ID
31.1.4 Create a Validation Check

Use this API to create a validation check.

**Name**  DME_PUB_VALIDATION_CHECK.CreateValidationCheck

**Signature:**

```sql
PROCEDURE CREATEVALIDATIONCHECK
(  P_API_VERSION IN VARCHAR2,
  P_INIT_MSG_LIST IN VARCHAR2 DEFAULT CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_COMMIT IN VARCHAR2 DEFAULT CDR_PUB_DEF_CONSTANTS.G_FALSE,
  P_VALIDATION_LEVEL IN NUMBER DEFAULT CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
  X_RETURN_STATUS OUT NOCOPY VARCHAR2,
  X_MSG_COUNT OUT NOCOPY NUMBER,
  X_MSG_DATA OUT NOCOPY VARCHAR2,
  PI_XFORMMAPCOLL IN OUT NOCOPY DME_XFORM_MAP_EX_COLL,
  PI_VCBATCHMODEOBJ IN CDR_BASE_OBJ_TYPE
);
```

**Parameters:**  This API has standard parameters (see "Standard Parameters" on page 2-5) as well as the following parameters:

- **PI_XFORMMAPCOLL** (Mandatory). This is a parameter of collection type DME_XFORM_MAP_EX_COLL. DME_XFORM_MAP_EX_COLL is a table of DME_XFORM_MAP_EX_OBJ_TYPE.
  
  This collection is prepared with both Table level and Column Level maps. Validation Checks Details are set at Table level mapping.
  
  The following attributes from DME_XFORM_MAP_EX_OBJ_TYPE are required for this API:
  
  - NAMING: Table type CDR_NAMING_VERSION_OBJ_TYPE
  
  For Table level mapping and Validation Checks Create, mandatory attributes are:
  
  - COMPANY_ID
  
  - NAME - it refers to Validation Checks name
  
  - OBJECT_TYPE_RC - Enter the value: '$OBJTYPES$XFORMMAP'.
  
  For Column Level mapping and Validation Checks Create, mandatory attributes are:
  
  - COMPANY_ID
  
  - OBJECT_TYPE_RC - Enter the value: '$OBJTYPES$XFORMMAP'.

- MAP_TYPE. Enter `$MAPTYPE$VC`

- XFORM_TYPE. Enter '$XFORMTYPE$DIRECT' for single source table and '$XFORMTYPE$JOIN' for multiple source table.

- PROGRAM_ID. If the validation check uses a custom program, enter the Program obj_id, else leave blank.

- PROGRAM_VER. If the validation check uses a custom program, enter the Program obj_ver, else leave blank.
- **PROGRAM_TYPE**. ‘CUSTOM’ for Custom program. Else leave blank. When creating Validation Checks using Custom Program, column level maps are not required in the collection parameter (PI_XFORMMAPCOLL).

- **AUTH_FLAG_RC**. Enter ‘$YESNO$YES’ to authorize Validation Checks listing to read blinded data. Otherwise leave blank. It defaults to ‘$YESNO$NO’.

- **OPERATION_TYPE**. ‘$OPER$CREATE’ for Validation Checks create.

- **MAP_ENTITY_COLL**. Its collection type DME_MAP_ENTITY_COLL which is table of DME_MAP_ENTITY_OBJ_TYPE type.
  
  For table level mapping, only source entities are required in MAP_ENTITY_COLL collection. Enter COMPANY_ID, DATAENTITY_ID, DATAENTITY_VER, MAP_RELATION as SOURCE, expression details (see Chapter 28.2, “Create or Modify an Expression,”) and OPERATION_TYPE as ‘$OPER$CREATE’.

  For column level mapping, both source and target entities are required in MAP_ENTITY_COLL collection. For source entities, enter company_id, dataentity_id, dataentity_ver, map_relation as SOURCE, expression details and operation_type as ‘$OPER$CREATE’ and for target entity, enter ALIAS, map_relation as TARGET and operation_type as ‘$OPER$CREATE’.

- **JOIN_COLL**. Required only for VC using multiple source tables. This is a collection of type CDR_DM_JOIN_OBJ_COLL. Set only in table level mapping.

  CDR_DM_JOIN_OBJ_COLL is table of type CDR_DM_JOIN_OBJ_TYPE.

  CDR_DM_JOIN_OBJ_TYPE is for Table Joins and set the source and target table ids. This object type has an attribute of collection type DM_JOIN_COL_OBJ_COLL for column joins.

  DM_JOIN_COL_OBJ_COLL is table of type CDR_DM_JOIN_COL_OBJ_TYPE. Table and Column related fields are required along with JOIN_OPERATOR_RC. For POSITION enter 1.

- **VC_DETAILS**. Table of dme_val_check_details_obj_type. Set only in table level mapping.

  COMPANY_ID, DISC_OPEN_STATE, DISCREPANCY_TEXT, PRIMARY_SOURCE_COLUMN_ID are mandatory. AUTO_CLOSE_FLAG,CATEGORY_ID and INITIAL_DISC_ACTION_ID are optional.

  * **DISC_OPEN_STATE**. Possible values are ‘$DISC_STATES$OPEN’ and ‘$DISC_STATES$CANDIDATE’.

  * **DISCREPANCY_TEXT**. Enter a text as comment for created discrepancies from Validation Checks.

  * **PRIMARY_SOURCE_COLUMN_ID**. Source Column OBJ_ID on which discrepancy is created.

  * **AUTO_CLOSE_FLAG**. Possible values are ‘$YESNO$YES’ and ‘$YESNO$NO’. Enter ‘$YESNO$YES’, if Validation Check can auto close the discrepancy.

  * **CATEGORY_ID**. Enter a valid Validation Check category Id from DME_CATEGORIES.
* INITIAL_DISC_ACTION_ID. : If discrepancies need DM review, enter 31 when DISC_OPEN_STATE='$DISC_STATES$CANDIDATE' or enter 32 when DISC_OPEN_STATE='$DISC_STATES$OPEN'.

- PI_VCBATCHMODELOBJ (Mandatory). This is a parameter of table type CDR_BASE_OBJ_TYPE that contains object attributes. Enter values to identify the Validation Checks Batch Model under which you want to create Validation Checks.

### 31.1.5 Update a Validation Check

Use this API to update a validation check.

**Name**  
DME_PUB_VALIDATION_CHECK.UpdateValidationCheck

**Signature:**

```sql
PROCEDURE UPDATEVALIDATIONCHECK
(P_API_VERSION IN VARCHAR2,
 P_INIT_MSG_LIST IN VARCHAR2 DEFAULT CDR_PUB_DEF_CONSTANTS.G_FALSE,
 P_COMMIT IN VARCHAR2 DEFAULT CDR_PUB_DEF_CONSTANTS.G_FALSE,
 P_VALIDATION_LEVEL IN NUMBER DEFAULT CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
 X_RETURN_STATUS OUT NOCOPY VARCHAR2,
 X_MSG_COUNT OUT NOCOPY NUMBER,
 X_MSG_DATA OUT NOCOPY VARCHAR2,
 PI_XFORMMAPCOLL IN OUT NOCOPY DME_XFORM_MAP_EX_COLL
);
```

**Parameters:**  
This API has standard parameters (see "Standard Parameters" on page 2-5) as well as the following parameters:

- PI_XFORMMAPCOLL (Mandatory). This is a parameter of collection type DME_XFORM_MAP_EX_COLL and DME_XFORM_MAP_EX_COLL is table of DME_XFORM_MAP_EX_OBJ_TYPE.  

This collection is prepared with both Table level and Column Level maps. Validation Check Details are set at Table level mapping.  

The attributes required for this API from DME_XFORM_MAP_EX_OBJ_TYPE are:

- For Table Level mapping, all the attributes are required. Name refers to Validation Checks name and enter OBJECT_TYPE_RC = '$OBJTYPES$XFORMMAP'.

- For Column Level mapping all the attributes are required. Enter OBJECT_TYPE_RC = '$OBJTYPES$XFORMMAP'.

- MAP_TYPE. Enter '$MAPTYPE$VC'.

- XFORM_TYPE. Enter '$XFORMTYPE$DIRECT' for single source table and '$XFORMTYPE$JOIN' for multiple source table.

- PROGRAM_ID. Enter Custom Program Obj_Id when Create Validation Checks using custom program. In other cases leave blank.

- PROGRAM_VER. Enter Custom Program Obj_Ver when Create Validation Checks using custom program. In other cases leave blank.

- PROGRAM_TYPE. Write 'CUSTOM' for Custom program, else leave blank. When creating Validation Checks using Custom Program, column level maps are not required in the collection parameter (PI_XFORMMAPCOLL).
- **AUTH_FLAG_RC.** Enter 'YESNOYES' to authorize Validation Checks listing to read blinded data. Otherwise leave blank. It defaults to 'YESNOSNO'.

- **OPERATION_TYPE.** 'OPERSMODIFY' for Validation Checks modify.

- **MAP_ENTITY_COLL.** It's collection type DME_MAP_ENTITY_COLL which is table of DME_MAP_ENTITY_OBJ_TYPE type. Leave blank, if you do not want to update anything.

  For table level, enter both source and target entities with entity identifiers, data_entity identifiers, map_relation as SOURCE or TARGET, expression details (see Chapter 28.2, "Create or Modify an Expression," for source and operation type as 'OPERSMODIFY'.

  For column level:

  a. Adding a new column, both source and target entities are required in MAP_ENTITY_COLL collection. For source entities, enter COMPANY_ID, DATAENTITY_ID, DATAENTITY_VER, MAP_RELATION as SOURCE, expression details (see Chapter 28.2, "Create or Modify an Expression," and operation_type as 'OPERSCREATE' and for target entity, enter ALIAS, map_relation as TARGET and operation_type as 'OPERSCREATE'.

  b. Updating a column, both source and target entities are required in MAP_ENTITY_COLL collection. For source entities, enter COMPANY_ID, MAP_ENTITY_ID, MAP_ENTITY_VER, DATAENTITY_ID, DATAENTITY_VER, MAP_RELATION as SOURCE, expression details (see Chapter 28.2, "Create or Modify an Expression," and operation_type as 'OPERSMODIFY' and for target entity, COMPANY_ID, MAP_ENTITY_ID, MAP_ENTITY_VER, DATAENTITY_ID, DATAENTITY_VER, entering ALIAS, MAP_RELATION as TARGET and operation_type as 'OPERSMODIFY'.

  c. Removing a column, both source and target entities are required in MAP_ENTITY_COLL collection. For both source and target entity, enter COMPANY_ID, MAP_ENTITY_ID, MAP_ENTITY_VER, DATAENTITY_ID, DATAENTITY_VER, MAP_RELATION as SOURCE or TARGET and OPERATION_TYPE as 'OPERSREMOVE'.

- **JOIN_COLL.** Leave blank, if you do not want to update anything. Required only for Validation Checks using multiple source tables. This is a collection of type CDR_DM_JOIN_OBJ_COLL. Set only in table level mapping.

  CDR_DM_JOIN_OBJ_COLL is table of type CDR_DM_JOIN_OBJ_TYPE.

  CDR_DM_JOIN_OBJ_TYPE is for Table Joins and set the source and target table ids. This object type has an attribute of collection type DM_JOIN_COL_OBJ_COLL for column joins.

  DM_JOIN_COL_OBJ_COLL is table of type CDR_DM_JOIN_COL_OBJ_TYPE. Table and Column related fields are required along with JOIN_OPERATOR_RC. For POSITION enter 1.

- **VC_DETAILS.** Table of DME_VAL_CHECK_DETAILS_OBJ_TYPE. Set only in table level mapping. COMPANY_ID,DISC_OPEN_STATE, DISCREPANCY_TEXT, PRIMARY_SOURCE_COLUMN_ID are mandatory. AUTO_CLOSE_FLAG, CATEGORY_ID and INITIAL_DISC_ACTION_ID are optional. OBJ_ID and OBJ_VER are required only in case of Validation Checks update.
* DISC_OPEN_STATE: Possible values are ’$DISC_STATES$OPEN’ and
‘$DISC_STATES$CANDIDATE’.
* DISCREPANCY_TEXT: Enter a text as comment for created discrepancies
from Validation Checks.
* PRIMARY_SOURCE_COLUMN_ID: Source Column OBJ_ID on which
discrepancy is created.
* AUTO_CLOSE_FLAG: Possible values are ’$YESNO$YES’ and
’$YESNO$NO’. Enter ’$YESNO$YES’, if Validation Checks can auto close
the discrepancy.
* CATEGORY_ID: Enter a valid Validation Checks category ID from DME_ CATEGORIES.
* INITIAL_DISC_ACTION_ID : If discrepancies need DM review, enter 31
when DISC_OPEN_STATE='$DISC_STATES$CANDIDATE' or enter 32
when DISC_OPEN_STATE='$DISC_STATES$OPEN'.

31.1.6 Install a Validation Check Batch

Use this API to install a validation check batch.

**Name**  DME_PUB_VALIDATION_CHECK.InstallValidationCheckBatch

**Signature:**

PROCEDURE INSTALLVALIDATIONCHECKBATCH
(P_API_VERSION IN VARCHAR2,
P_INIT_MSG_LIST IN VARCHAR2 DEFAULT CDR_PUB_DEF_CONSTANTS.G_FALSE,
P_COMMIT IN VARCHAR2 DEFAULT CDR_PUB_DEF_CONSTANTS.G_FALSE,
P_VALIDATION_LEVEL IN NUMBER DEFAULT CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
X_RETURN_STATUS OUT NOCOPY VARCHAR2,
X_MSG_COUNT OUT NOCOPY NUMBER,
X_MSG_DATA OUT NOCOPY VARCHAR2,
PI_BATCHMAPCOMPID IN CDR_NAMINGS.COMPANY_ID%TYPE,
PI_BATCHMAPOBJID IN CDR_NAMINGS.OBJ_ID%TYPE,
PI_BATCHMAPOBJVER IN CDR_NAMING_VERSIONS.OBJ_VER%TYPE,
PI_CONTEXT IN VARCHAR2 DEFAULT NULL,
PI_TESTMODE IN VARCHAR2 DEFAULT NULL,
PO_JOBID OUT NOCOPY VARCHAR2
);

**Parameters:**  This API has standard parameters (see ”Standard Parameters” on
page 2-5) as well as the following parameters:

- **PI_BATCHMAPCOMPID** (Mandatory). Enter COMPANY_ID of Validation
Checks Batch Map.
- **PI_BATCHMAPOBJID** (Mandatory). Enter OBJ_ID of Validation Checks Batch
Map.
- **PI_BATCHMAPOBJVER** (Mandatory). Enter OBJ_VER of Validation Checks
Batch Map.
- **PI_CONTEXT** (Optional). Enter values as $LIFECYCLE$DEV, $LIFECYCLE$QC
or $LIFECYCLE$PROD. If entered as null, system tries to find out the SYSTEM_ CONTEXT. If SYITEM_Context is not set, it defaults to $LIFECYCLE$DEV.
- **PI_TESTMODE** (Optional). Leave blank
- **PO_JOBID** (Optional). Installation of Validation Checks Batch submits a job. So this output parameter returns a JOB_ID for Validation Checks Batch installation.

### 31.1.7 Submit a Validation Check Batch

Use this API to submit a validation check batch.

**Name**  
DME_PUB_VALIDATION_CHECK.DeleteFlag

**Signature:**

```sql
FUNCTION SUBMITVALIDATIONCHECKBATCH
(P_API_VERSION IN VARCHAR2,
P_INIT_MSG_LIST IN VARCHAR2 DEFAULT CDR_PUB_DEF_CONSTANTS.G_FALSE,
P_COMMIT IN VARCHAR2 DEFAULT CDR_PUB_DEF_CONSTANTS.G_FALSE,
P_VALIDATION_LEVEL IN NUMBER DEFAULT CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
X_RETURN_STATUS OUT NOCOPY VARCHAR2,
X_MSG_COUNT OUT NOCOPY NUMBER,
X_MSG_DATA OUT NOCOPY VARCHAR2,
PI_VCBATCHMODELCOMPANYID IN CDR_NAMINGS.COMPANY_ID%TYPE,
PI_VCBATCHMODELOBJID IN CDR_NAMINGS.OBJ_ID%TYPE,
PI_VCBATCHMODELOBJVER IN CDR_NAMING_VERSIONS.OBJ_VER%TYPE,
PI_SUBMISSION_TYPE IN VARCHAR2 DEFAULT '$SUBMISSTYPES$IMMEDIATE',
PI_CONTEXT IN VARCHAR2 DEFAULT '$LIFECYCLE$DEV',
PI_SCHED_START_TS IN DATE DEFAULT NULL,
PI_SCHED_END_TS  IN DATE DEFAULT NULL,
PI_SCHED_REPEAT_INTERVAL IN VARCHAR2 DEFAULT NULL,
PI_SCHED_REPEAT_LIST IN VARCHAR2 DEFAULT NULL,
PI_FORCE_EXECUTION_FLAG_RC IN VARCHAR2 DEFAULT '$YESNO$NO',
PI_RUN_MODE_RC IN VARCHAR2)
RETURN CDR_JOBS.JOB_ID%TYPE;
```

**Parameters:**  
This API has standard parameters (see "Standard Parameters" on page 2-5) as well as the following parameters:

- **PI_VCBATCHMODELCOMPANYID** (Mandatory). Company ID of Validation Checks Batch Model.

- **PI_VCBATCHMODELOBJID** (Mandatory). OBJ_ID of Validation Checks Batch Model.

- **PI_VCBATCHMODELOBJVER** (Mandatory). OBJ_VER of Validation Checks Batch Model.

- **PI_RUN_MODE_RC** (Mandatory). Enter '$RUNMODES$FULL' or '$RUNMODES$INCREMENT'.

- **PI_SUBMISSION_TYPE**. Enter '$SUBMISSTYPES$IMMEDIATE' or '$SUBMISSTYPES$SCHEDULED' or '$SUBMISSTYPES$DEFERRED'.

- **PI_CONTEXT**. Lifecycle context values like $LIFECYCLE$DEV, $LIFECYCLE$QC or $LIFECYCLE$PROD.

- **PI_SCHED_START_TS**. Enter start time if submission is scheduled/deferred.

- **PI_SCHED_END_TS**. Enter end time if submission is scheduled.

- **PI_SCHED_REPEAT_INTERVAL**. Enter interval time if submission is scheduled.

- **PI_SCHED_REPEAT_LIST**. Enter if submission is scheduled. Accepted values are $SUBMISSTYPES$HOURLY, $SUBMISSTYPES$DAILY, $SUBMISSTYPES$WEEKLY, $SUBMISSTYPES$MONTHLY.
Create and Modify Validation Checks and Batches

- **PI_FORCE_EXECUTION_FLAG_RC.** Enter '$YESNO$NO' or '$YESNO$YES'.
  This API returns the Job Id.

### 31.1.8 Check In a Validation Check Batch

Use this API to check in a validation check batch.

**Name** DME_PUBVALIDATION_CHECK.CheckinValidationCheckBatch

**Signature:**

PROCEDURE CHECKINVALIDATIONCHECKBATCH
  (P_API_VERSION IN NUMBER,
   P_INIT_MSG_LIST IN VARCHAR2 DEFAULT CDR_PUB_DEF_CONSTANTS.G_FALSE,
   P_COMMIT IN VARCHAR2 DEFAULT CDR_PUB_DEF_CONSTANTS.G_FALSE,
   P_VALIDATION_LEVEL IN NUMBER DEFAULT CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
   X_RETURN_STATUS OUT NOCOPY VARCHAR2,
   X_MSG_COUNT OUT NOCOPY NUMBER,
   X_MSG_DATA OUT NOCOPY VARCHAR2,
   PIO_NAMING IN OUT NOCOPY CDR_NAMING_VERSION_OBJ_TYPE,
   PI_COMMENT IN VARCHAR2,
  );

**Parameters:** This API has standard parameters (see "Standard Parameters" on page 2-5) as well as the following parameters:

- **PIO_NAMING (Mandatory).** This is a parameter of table type CDR_NAMING_VERSION_OBJ_TYPE that contains object attributes. Enter values to identify the Validation Checks Batch that you want to check in.
  The following attributes are required:
  - COMPANY_ID
  - OBJ_ID
  - OBJ_VER

- **PI-comment (Optional).** Enter the reason you are checking in the Validation Checks Batch.

### 31.1.9 Check Out a Validation Check Batch

Use this API to check out a validation check batch.

**Name** DME_PUBVALIDATION_CHECK.CheckoutValidationCheckBatch

**Signature:**

PROCEDURE checkoutValidationCheckBatch
  (P_API_VERSION IN NUMBER,
   P_INIT_MSG_LIST IN VARCHAR2 DEFAULT CDR_PUB_DEF_CONSTANTS.G_FALSE,
   P_COMMIT IN VARCHAR2 DEFAULT CDR_PUB_DEF_CONSTANTS.G_FALSE,
   P_VALIDATION_LEVEL IN NUMBER DEFAULT CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
   X_RETURN_STATUS OUT NOCOPY VARCHAR2,
   X_MSG_COUNT OUT NOCOPY NUMBER,
   X_MSG_DATA OUT NOCOPY VARCHAR2,
   PIO_NAMING IN OUT NOCOPY CDR_NAMING_VERSION_OBJ_TYPE,
   PI_COMMENT IN VARCHAR2,
  );
Parameters: This API has standard parameters (see "Standard Parameters" on page 2-5) as well as the following parameters:

- **PIO_NAMING** (Mandatory). This is a parameter of table type CDR_NAMING_VERSION_OBJ_TYPE that contains object attributes. Enter values to identify the Validation Checks Batch that you want to check in.

  The following attributes are required:
  - COMPANY_ID
  - OBJ_ID
  - OBJ_VER

- **PI_TAB_OBJ_ID** (Mandatory). This input parameter is an object id for a namings object of type $OBJTYPES$TABLEREF (table instance).

- **PI_COMMENT** (Optional). Enter the reason you are checking in the Validation Check Batch.

31.1.10 Undo Checkout For a Validation Check Batch

Use this API to undo check out for a Validation Check Batch.

**Name** DME_PUB_VALIDATION_CHECK.UncheckValidationCheckBatch

**Signature:**

```sql
PROCEDURE DELETEFLAG
(P_API_VERSION IN NUMBER,
P_INIT_MSG_LIST IN VARCHAR2 DEFAULT CDR_PUB_DEF_CONSTANTS.G_FALSE,
P_COMMIT IN VARCHAR2 DEFAULT CDR_PUB_DEF_CONSTANTS.G_FALSE,
P_VALIDATION_LEVEL IN NUMBER DEFAULT CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
X_RETURN_STATUS OUT NOCOPY VARCHAR2,
X_MSG_COUNT OUT NOCOPY NUMBER,
X_MSG_DATA OUT NOCOPY VARCHAR2,
PIO_NAMING IN OUT NOCOPY CDR_NAMING_VERSION_OBJ_TYPE,
PI_COMMENT IN VARCHAR2);
```

Parameters: This API has standard parameters (see "Standard Parameters" on page 2-5) as well as the following parameters:

- **PIO_NAMING** (Mandatory). This is a parameter of table type CDR_NAMING_VERSION_OBJ_TYPE that contains object attributes. Enter values to identify the Validation Check Batch that you want to check in.

  The following attributes are required:
  - COMPANY_ID
  - OBJ_ID
  - OBJ_VER

- **PI_COMMENT** (Optional). Enter the reason you are checking in the Validation Checks Batch.

31.1.11 Update Validation Status of a Validation Check Batch

Use this API to update validation status of a Validation Check Batch.
Name  DME_PUB_VALIDATION_CHECK.UpdateValStatus

Signature:

PROCEDURE UPDATEVALSTATUS
( P_API_VERSION IN NUMBER,
P_INIT_MSG_LIST IN VARCHAR2 DEFAULT CDR_PUB_DEF_CONSTANTS.G_FALSE,
P_COMMIT IN VARCHAR2 DEFAULT CDR_PUB_DEF_CONSTANTS.G_FALSE,
P_VALIDATION_LEVEL IN NUMBER DEFAULT CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
X_RETURN_STATUS OUT NOCOPY VARCHAR2,
X_MSG_COUNT OUT NOCOPY NUMBER,
X_MSG_DATA OUT NOCOPY VARCHAR2,
PIO_NAMING IN OUT NOCOPY CDR_NAMING_VERSION_OBJ_TYPE,
PI_TESTMODE IN VARCHAR2 DEFAULT NULL
);

Parameters:  This API has standard parameters (see "Standard Parameters" on page 2-5) as well as the following parameters:

■ PIO_NAMING (Mandatory). This is a parameter of table type CDR_NAMING_VERSION_OBJ_TYPE that contains object attributes. Enter values to identify the Validation Checks Batch that you want to update validation status.

All the attributes are required.

■ PI_TESTMODE (Optional). Leave blank.

31.12 Upgrade a Validation Check Batch

Use this API to upgrade a validation check batch.

Name  DME_PUB_VALIDATION_CHECK.UpgradeValidationCheckBatch

Signature:

PROCEDURE UPGRADEVALIDATIONCHECKBATCH
(P_API_VERSION IN VARCHAR2,
P_INIT_MSG_LIST IN VARCHAR2 DEFAULT CDR_PUB_DEF_CONSTANTS.G_FALSE,
P_COMMIT IN VARCHAR2 DEFAULT CDR_PUB_DEF_CONSTANTS.G_FALSE,
P_VALIDATION_LEVEL IN NUMBER DEFAULT CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
X_RETURN_STATUS OUT NOCOPY VARCHAR2,
X_MSG_COUNT OUT NOCOPY NUMBER,
X_MSG_DATA OUT NOCOPY VARCHAR2,
PI_BATCHMAPCOMPID IN CDR_NAMINGS.COMPANY_ID%TYPE,
PI_BATCHMAPOBJID IN CDR_NAMINGS.OBJ_ID%TYPE,
PI_BATCHMAPOBJVER IN CDR_NAMING_VERSIONS.OBJ_VER%TYPE
);

Parameters:  This API has standard parameters (see "Standard Parameters" on page 2-5) as well as the following parameters:

■ PI_BATCHMAPCOMPID (Mandatory). Enter COMPANY_ID of Validation Checks Batch Map.

■ PI_BATCHMAPOBJID (Mandatory). Enter OBJ_ID of Validation Checks Batch Map.

■ PI_BATCHMAPOBJVER (Mandatory). Enter OBJ_VER of Validation Checks Batch Map.
## 31.1.13 Remove Validation Check(s)

Use this API to remove validation check(s).

**Name**  
DME_PUB_VALIDATION_CHECK.RemoveValidationChecks

**Signature:**

```sql
PROCEDURE REMOVEVALIDATIONCHECKS
(P_API_VERSION IN VARCHAR2,
 P_INIT_MSG_LIST IN VARCHAR2 DEFAULT CDR_PUB_DEF_CONSTANTS.G_FALSE,
 P_COMMIT IN VARCHAR2 DEFAULT CDR_PUB_DEF_CONSTANTS.G_FALSE,
 P_VALIDATION_LEVEL IN NUMBER DEFAULT CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
 X_RETURN_STATUS OUT NOCOPY VARCHAR2,
 X_MSG_COUNT OUT NOCOPY NUMBER,
 X_MSG_DATA OUT NOCOPY VARCHAR2,
 PIO_VC_COLL IN OUT NOCOPY CDR_NAMING_LIST_COLL);
```

**Parameters:**  
This API has standard parameters (see “Standard Parameters” on page 2-5) as well as the following parameters:

- **PIO_VC_COLL** (Mandatory). This is a parameter of collection type CDR_NAMING_LIST_COLL that contains CDR Naming Version attributes.
  
The attributes required for this API are:
  
  - COMPANY_ID
  - OBJ_ID
  - OBJ_VER

## 31.1.14 Enable or Disable Validation Checks

Use this API to enable or disable Validation Checks.

**Name**  
DME_PUB_VALIDATION_CHECK.EnableDisableValidationChecks

**Signature:**

```sql
PROCEDURE ENABLEDISABLEVALIDATIONCHECKS
(P_API_VERSION IN VARCHAR2,
 P_INIT_MSG_LIST IN VARCHAR2 DEFAULT CDR_PUB_DEF_CONSTANTS.G_FALSE,
 P_COMMIT IN VARCHAR2 DEFAULT CDR_PUB_DEF_CONSTANTS.G_FALSE,
 P_VALIDATION_LEVEL IN NUMBER DEFAULT CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
 X_RETURN_STATUS OUT NOCOPY VARCHAR2,
 X_MSG_COUNT OUT NOCOPY NUMBER,
 X_MSG_DATA OUT NOCOPY VARCHAR2,
 PI_VC_MAP_COLL IN CDR_BASE_OBJ_COLL,
 PI_ENABLE_FLAG IN VARCHAR2);
```

**Parameters:**  
This API has standard parameters (see “Standard Parameters” on page 2-5) as well as the following parameters:

- **PI_VC_MAP_COLL** (Mandatory). This is a parameter of collection type CDR_BASE_OBJ_COLL that contains CDR base object attributes. The following attributes are required: COMPANY_ID, OBJ_ID, OBJ_VER for Validation Checks maps.
- **PI_ENABLE_FLAG** (Mandatory). Enter 'YESNO$YES' for enabling and 'YESNO$NO' for disabling.

### 31.1.15 Reorder Validation Checks

Use this API to re-order validation checks.

**Name** DME_PUB_VALIDATION_CHECK.ReorderValidationChecks

**Signature:**

```sql
PROCEDURE REORDERVALIDATIONCHECKS
(P_API_VERSION IN VARCHAR2,
P_INIT_MSG_LIST IN VARCHAR2 DEFAULT CDR_PUB_DEF_CONSTANTS.G_FALSE,
P_COMMIT IN VARCHAR2 DEFAULT CDR_PUB_DEF_CONSTANTS.G_FALSE,
P_VALIDATION_LEVEL IN NUMBER DEFAULT CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
X_RETURN_STATUS OUT NOCOPY VARCHAR2,
X_MSG_COUNT OUT NOCOPY NUMBER,
X_MSG_DATA OUT NOCOPY VARCHAR2,
PI_VC_MAP_COLL IN OUT NOCOPY DME_XFORM_MAP_COLL
);
```

**Parameters:** This API has standard parameters (see "Standard Parameters" on page 2-5) as well as the following parameters:

- **PI_VC_MAP_COLL** (Mandatory). This is a parameter of collection type DME_XFORM_MAP_COLL which is a table of DME_XFORM_MAP_OBJ_TYPE type. The attributes required for this API from DME_XFORM_MAP_OBJ_TYPE are:
  - COMPANY_ID
  - OBJ_ID
  - OBJ_VER
  - EXECUTION_ORDER_NUMBER
This is a public interface for all operations related to Transformation Maps, including creation, deletion, modification, and checking in and out of these objects.

This section includes the describes the details of the Public APIs for transformation module DME_PUB_XFORM_MAP.

### 32.1 Create and Modify Transformation Maps

This section contains the following public APIs:

- Section 32.1.1, "Create Transformation Maps"
- Section 32.1.2, "Modify Transformation Maps"
- Section 32.1.3, "Mark Table Maps Not Used"
- Section 32.1.4, "Mark Column Maps Not Used"
- Section 32.1.5, "Check In Transformation Maps"
- Section 32.1.6, "Check Out Transformation Maps"
- Section 32.1.7, "Undo Checkout Transformation Map"
- Section 32.1.8, "Auto Map Tables"
- Section 32.1.9, "Accept Table Mappings"
- Section 32.1.10, "Auto Map Columns"
- Section 32.1.11, "Accept Column Mappings"
- Section 32.1.12, "Upgrade Transformation Map"
- Section 32.1.13, "Install Transformation Map"
- Section 32.1.14, "Remove Transformation Map"
- Section 32.1.15, "Validate Transformation Maps"
- Section 32.1.16, "Update Validation Status"
- Section 32.1.17, "Execute Transformation Map"
- Section 32.1.18, "Create Staging Table"
- Section 32.1.19, "Validate Expression"
32.1.1 Create Transformation Maps

Use this API to create model-, table- and column-level transformation maps. You run this API multiple times to create the model-level mapping first, then each table mapping, and then each column mapping.

**Name**  DME_PUB_XFORM_MAP.CreateTransformationMap

**Signature:**

```sql
PROCEDURE CREATETRANSFORMATIONMAP,
(P_API_VERSION IN VARCHAR2,
 P_INIT_MSG_LIST IN VARCHAR2 DEFAULT CDR_PUB_DEF_CONSTANTS.G_FALSE,
 P_COMMIT IN VARCHAR2 DEFAULT CDR_PUB_DEF_CONSTANTS.G_FALSE,
 P_VALIDATION_LEVEL IN NUMBER DEFAULT CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
 X_RETURN_STATUS OUT NOCOPY VARCHAR2,
 X_MSG_COUNT OUT NOCOPY NUMBER,
 X_MSG_DATA OUT NOCOPY VARCHAR2,
 PIO_XFORMMAP IN OUT NOCOPY DME_XFORM_MAP_EX_OBJ_TYPE
);
```

**Parameters:** This API has standard parameters (see "Standard Parameters" on page 2-5) as well as the following parameters:

- **OBJECT_TYPE_RC** - Enter the OBJECT_TYPE_RC as '${OBJTYPES$XFORMMAP}'
- **NAMESPACE_OBJ_ID** - Enter the object ID of the parent in which you want to create transformation map. Parent object ID will be study ID, model map ID and table map ID for creating a model, table and column map respectively.
- **NAMESPACE_OBJ_VER** - Enter the parent object version in which you want to create transformation map. Parent object ver will be study ver, model map ver and table map ver for creating a model, table and column map respectively.
- **PIO_XFORMMAP** (Mandatory). This is a parameter of DME_XFORM_MAP_EX_OBJ_TYPE object type. The attributes required for this API are:
  - **MAP_TYPE** - Enter Map_Type as '${MAPTYPE$DEFAULT' for transformation at model, table and column level.
  - **XFORM_TYPE** - Map_Type should be passed as '${XFORMTYPE$DIRECT' for model and column maps.
    For table level maps, it should have any one of following values based of table map type:
    - ${XFORMTYPE$DIRECT
    - ${XFORMTYPE$UNION
    - ${XFORMTYPE$JOIN
    - ${XFORMTYPE$CUSTOM
    - ${XFORMTYPE$PIVOT
    - ${XFORMTYPE$UNPIVOT
  - **PROGRAM_ID** - Enter Program Obj_Id while creating a table level map of transformation type as '${XFORMTYPE$CUSTOM', otherwise, leave blank.
  - **PROGRAM_VER** - Enter Program Obj_Ver while creating a table level map of transformation type as '${XFORMTYPE$CUSTOM', otherwise, leave blank.
– **PIVOT_COLUMN_ID** - Enter Object Id of the pivot column while creating a table transformations of pivot type, otherwise, leave blank.

– **PIVOT_COLUMN_VER** - Enter Object Version of the pivot column while creating a table transformations of pivot type, otherwise, leave blank.

– **OPERATION_TYPE** - Enter operation type as '$OPER$CREATE'.

– **MAP_ENTITY_COLL** - This attribute is a collection of DME_MAP_ENTITY_OBJ_TYPE object type attributes. The following attributes are required:

  * **DATA_ENTITY_ID** - For transformation map at model level, enter Source Data Model Object Id for Source Map Entity and Target Data Model Object ID for Target Map Entity.

For transformation map at table level, enter Source Table Definition Object Id for Source Map Entity and Target Table Definition Object ID for Target Map Entity.

For transformation map at column level, enter Source Column Object Id for Source Map Entity and Target Column Object ID for Target Map Entity.

  * **DATA_ENTITY_VER** - Enter Object Version correpsonding to the object selected for DATA_ENTITY_ID population.

  * **MAP_RELATION** - Map_relation should be passed as 'SOURCE' for source map entities and 'TARGET' for target map entities.

  * **EXPR_OBJ_TYPE** - This parameter is a object type of DME_XFORM_EXPR_OBJ_TYPE. Pass the expression details for source filter for source tables in case of table level transformations. Refer to the following section Chapter 28.2, "Create or Modify an Expression," for more details.

  * **OPERATION_TYPE** - The operation type should be passed as '$OPER$CREATE' for all source and target entities.

– **JOIN_COLL** - This is a parameter of collection type CDR_DM_JOIN_OBJ_COLL and CDR_DM_JOIN_OBJ_COLL is table of CDR_DM_JOIN_OBJ_TYPE object type. This collection is required for creating table level transformations of 'Join' Type. The following attributes are required for join transformations:

  * **COMPANY_ID** - Company Id

  * **TAB_COMPANY_ID** -Table Company Id

  * **TAB_OBJ_ID** - Table Obj Id

  * **TAB_ALIAS** - Table Alias

  * **FK_TAB_COMPANY_ID** - Second table company Id which is joined with first table

  * **FK_TAB_OBJ_ID** - Second table obj Id which is joined with first table

  * **FK_TAB_ALIAS** - Alias for second table

  * **TD_OUTERJOIN_RC** - Set to '$YESNO$YES' if outer joined defined on first table, otherwise '$YESNO$NO'.

  * **FK_TD_OUTERJOIN_RC** - Set to '$YESNO$YES' if outer joined defined on second table, otherwise '$YESNO$NO'.

  * **DM_JOIN_COL_OBJ_COLL** - This is a parameter of collection type CDR_DM_JOIN_COL_OBJ_COLL and CDR_DM_JOIN_COL_OBJ_COLL is table of CDR_DM_JOIN_COL_OBJ_TYPE object type. This collection is required for populating the join conditions. The attributes required for this API are:
Create and Modify Transformation Maps

32-4
Oracle Life Sciences Data Hub Application Programming Interface Guide

* COMPANY_ID - Company Id
* TAB_COMPANY_ID - Table Company Id
* TAB_OBJ_ID - Table Obj Id
* TAB_ALIAS - Table Alias
* TAB_COL_COMPANY_ID - Joined Table Column Company Id
* TAB_COL_OBJ_ID - Table Column Obj Id
* FK_TAB_COMPANY_ID - Second table company Id which is joined with first table
* FK_TAB_OBJ_ID - Second table obj Id which is joined with first table
* FK_TAB_ALIAS - Alias for second table
* FK_TAB_COL_COMPANY_ID - Table Column Company Id from second table which is joined with first table
* FK_TAB_COL_OBJ_ID - Table Column Obj Id from second table which is joined with first table
* POSITION - Sequence of join condition. The sequence should start for first join condition from 1.
* JOIN_OPERATOR_RC - This refers to type of join condition. This can have the following values:
  - $JOINOPER$EQUALS
  - $JOINOPER$EQUALS
  - $JOINOPER$GREATER
  - $JOINOPER$GREATEREQUAL
  - $JOINOPER$LESS
  - $JOINOPER$LESSEQUAL
  - $JOINOPER$NOTEQUALS

32.1.2 Modify Transformation Maps

Use this API to modify model, table and column level transformation maps.

**Name**  DME_PUB_XFORM_MAP.ModifyTransformationMap

**Signature:**

PROCEDURE MODIFYTRANSFORMATIONMAP
(P_API_VERSION IN VARCHAR2,
 P_INIT_MSG_LIST IN VARCHAR2 DEFAULT CDR_PUB_DEF_CONSTANTS.G_FALSE,
 P_COMMIT IN VARCHAR2 DEFAULT CDR_PUB_DEF_CONSTANTS.G_FALSE,
 P_VALIDATION_LEVEL IN NUMBER DEFAULT CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
 X_RETURN_STATUS OUT NOCOPY VARCHAR2,
 X_MSG_COUNT OUT NOCOPY NUMBER,
 X_MSG_DATA OUT NOCOPY VARCHAR2,
 PIO_XFORMMAP IN OUT NOCOPY DME_XFORM_MAP_EX_OBJ_TYPE,
);

**Parameters:**  This API has standard parameters (see "Standard Parameters" on page 2-5) as well as the following parameters:
Create and Modify Transformation Maps

**PIO_XFORMMAP** (Mandatory). This is a parameter of DME_XFORM_MAP_EX_OBJ_TYPE object type. The attributes required for this API are:

- **MAP_TYPE** - Enter Map_Type of the transformation map. It can be specified as '$MAPTYPE$DEFAULT'.
- **XFORM_TYPE** - Enter map_Type of transformation map being modified. For table map, same of modified map type can be:
  * $XFORMTYPE$ DIRECT
  * $XFORMTYPE$UNION
  * $XFORMTYPE$JOIN
  * $XFORMTYPE$CUSTOM
  * $XFORMTYPE$PIVOT
  * $XFORMTYPE$UNPIVOT
- **PROGRAM_ID** - Enter the program object id, if table level transformation of custom type being modified, otherwise NULL.
- **PROGRAM_VER** - Enter the program object version, if table level transformation of custom type being modified, otherwise NULL.
- **PIVOT_COLUMN_ID** - Enter Object Id of a Column, if table level pivot transformations being modified, otherwise NULL.
- **PIVOT_COLUMN_VER** - Enter Object Version of a Column, if table level pivot transformations being modified, otherwise NULL.
- **OPERATION_TYPE** - Enter operation type as '$OPER$MODIFY'.
- **MAP_ENTITY_COLL** - This attribute is a collection of DME_MAP_ENTITY_OBJ_TYPE object type attributes. All attributes are required including identifying attributes along with required modifications. The operation_type should be passed as '$OPER$CREATE', '$OPER$MODIFY' or '$OPER$REMOVE' depending upon the modification required for a map entity.

The EXPR_OBJ_TYPE is one of the attribute in DME_MAP_ENTITY_OBJ_TYPE object type. This is a object type of DME_XFORM_EXPR_OBJ_TYPE. Pass the expression details for source filter for source tables in case of table level transformations. Refer to the following section Chapter 28.2, "Create or Modify an Expression," for more details.

**JOIN_COLL** - This is a parameter of collection type CDR_DM_JOIN_OBJ_COLL and CDR_DM_JOIN_OBJ_COLL is table of CDR_DM_JOIN_OBJ_TYPE object type. This collection is required for modifying table level transformations of 'Join' Type.

The attributes of CDR_DM_JOIN_OBJ_TYPE required for this API, including the modifications which are required in the join condition, are:

- **COMPANY_ID**
- **OBJ_ID**
- **OBJ_VER**
- **TABLE_NAME**
- **FK_TABLE_NAME**
- **TAB_COMPANY_ID**
Create and Modify Transformation Maps

- TAB_OBJ_ID
- TAB_ALIAS
- TAB_MAP_ENTITY_ID
- FK_TAB_COMPANY_ID
- FK_TAB_OBJ_ID
- FK_TAB_ALIAS
- FK_TAB_MAP_ENTITY_ID
- TD_OUTERJOIN_RC
- FK_TD_OUTERJOIN_RC
- DM_JOIN_COL_OBJ_COLL

The DM_JOIN_COL_OBJ_COLL attribute is a parameter of collection type CDR_DM_JOIN_COL_OBJ_COLL, and CDR_DM_JOIN_OBJ_COLL is table of CDR_DM_JOIN_COL_OBJ_TYPE object type.

The attributes of CDR_DM_JOIN_COL_OBJ_TYPE required for this API, including the modifications which are required in the join condition, are:

- COMPANY_ID
- DM_JOIN_OBJ_ID
- DM_JOIN_OBJ_VER
- TAB_COMPANY_ID
- TAB_OBJ_ID
- TAB_ALIAS
- TABLE_NAME
- TAB_COL_COMPANY_ID
- TAB_COL_OBJ_ID
- TABLE_COLUMN_NAME
- FK_TAB_COMPANY_ID
- FK_TAB_OBJ_ID
- FK_TAB_ALIAS
- FK_TABLE_NAME
- FK_TAB_COL_COMPANY_ID
- FK_TAB_COL_OBJ_ID
- FK_TABLE_COL_NAME
- POSITION
- JOIN_OPERATOR_RC

The JOIN_OPERATOR_RC refers to type of join condition. This can have the following values:

- $JOINOPER$EQUALS
- $JOINOPER$EQUALS
32.1.3 Mark Table Maps Not Used

Use this API to mark table mappings as 'Not Used'.

**Name**  
DME_PUB_XFORM_MAP.MarkTableMapNotUsed

**Signature:**

```sql
PROCEDURE MARKTABLEMAPNOTUSED
  (P_API_VERSION IN VARCHAR2,
   P_INIT_MSG_LIST IN VARCHAR2 DEFAULT CDR_PUB_DEF_CONSTANTS.G_FALSE,
   P_COMMIT IN VARCHAR2 DEFAULT CDR_PUB_DEF_CONSTANTS.G_FALSE,
   P_VALIDATION_LEVEL IN NUMBER DEFAULT CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
   X_RETURN_STATUS OUT NOCOPY VARCHAR2,
   X_MSG_COUNT OUT NOCOPY NUMBER,
   X_MSG_DATA OUT NOCOPY VARCHAR2,
   PI_MAPOBJID IN CDR_NAMING_VERSIONS.OBJ_ID%TYPE,
   PI_MAPOBJVER IN CDR_NAMING_VERSIONS.OBJ_VER%TYPE
  );
```

**Parameters:**  
This API has standard parameters (see “Standard Parameters” on page 2-5) as well as the following parameters:

- **PI_MAPOBJID** (Mandatory). Enter object Id of Table Transformation Map.
- **PI_MAPOBJVER** (Mandatory). Enter object Version of Table Transformation Map.

32.1.4 Mark Column Maps Not Used

Use this API to mark column mappings as 'Not Used'.

**Name**  
DME_PUB_XFORM_MAP.MarkColumnMapNotUsed

**Signature:**

```sql
PROCEDURE MARKCOLUMNMAPNOTUSED
  (P_API_VERSION IN VARCHAR2,
   P_INIT_MSG_LIST IN VARCHAR2 DEFAULT CDR_PUB_DEF_CONSTANTS.G_FALSE,
   P_COMMIT IN VARCHAR2 DEFAULT CDR_PUB_DEF_CONSTANTS.G_FALSE,
   P_VALIDATION_LEVEL IN NUMBER DEFAULT CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
   X_RETURN_STATUS OUT NOCOPY VARCHAR2,
   X_MSG_COUNT OUT NOCOPY NUMBER,
   X_MSG_DATA OUT NOCOPY VARCHAR2,
   PI_MAPOBJID IN CDR_NAMING_VERSIONS.OBJ_ID%TYPE,
   PI_MAPOBJVER IN CDR_NAMING_VERSIONS.OBJ_VER%TYPE
  );
```

**Parameters:**  
This API has standard parameters (see “Standard Parameters” on page 2-5) as well as the following parameters:

- **PI_MAPOBJID** (Mandatory). Enter object Id of Column Transformation Map.
32.1.5 Check In Transformation Maps

Use this API to check in the given transformation map.

**Name**  DME_PUB_XFORM_MAP.CheckinTransformationMap

**Signature:**

```sql
PROCEDURE CHECKINTRANSFORMATIONMAP
    (P_API_VERSION IN VARCHAR2,
     P_INIT_MSG_LIST IN VARCHAR2 DEFAULT CDR_PUB_DEF_CONSTANTS.G_FALSE,
     P_COMMIT IN VARCHAR2 DEFAULT CDR_PUB_DEF_CONSTANTS.G_FALSE,
     P_VALIDATION_LEVEL IN NUMBER DEFAULT CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
     X_RETURN_STATUS OUT NOCOPY VARCHAR2,
     X_MSG_COUNT OUT NOCOPY NUMBER,
     X_MSG_DATA OUT NOCOPY VARCHAR2,
     PIO_NAMING IN OUT NOCOPY CDR_NAMING_VERSION_OBJ_TYPE
    );
```

**Parameters:** This API has standard parameters (see "Standard Parameters" on page 2-5) as well as the following parameters:

- **PIO_NAMING** (Mandatory). This is a parameter of table type CDR_NAMING_VERSION_OBJ_TYPE that contains object attributes. Enter values to identify the transformation map that you want to check in. The following attributes are required:
  - COMPANY_ID
  - OBJ_ID
  - OBJ_VER
  - NAMESPACE_OBJ_ID
  - NAMESPACE_OBJ_VER
  - OBJECT_VERSION_NUMBER

32.1.6 Check Out Transformation Maps

Use this API to check out the given transformation map.

**Name**  DME_PUB_XFORM_MAP.CheckoutTransformationMap

**Signature:**

```sql
PROCEDURE CHECKOUTTRANSFORMATIONMAP
    (P_API_VERSION IN VARCHAR2,
     P_INIT_MSG_LIST IN VARCHAR2 DEFAULT CDR_PUB_DEF_CONSTANTS.G_FALSE,
     P_COMMIT IN VARCHAR2 DEFAULT CDR_PUB_DEF_CONSTANTS.G_FALSE,
     P_VALIDATION_LEVEL IN NUMBER DEFAULT CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
     X_RETURN_STATUS OUT NOCOPY VARCHAR2,
     X_MSG_COUNT OUT NOCOPY NUMBER,
     X_MSG_DATA OUT NOCOPY VARCHAR2,
     PIO_NAMING IN OUT NOCOPY CDR_NAMING_VERSION_OBJ_TYPE
    );
```
Parameters: This API has standard parameters (see “Standard Parameters” on page 2-5) as well as the following parameters:

- **PIO_NAMING** (Mandatory). This is a parameter of table type CDR_NAMING_VERSION_OBJ_TYPE that contains object attributes. Enter values to identify the transformation map.

  The attributes for this API are:
  - COMPANY_ID
  - OBJ_ID
  - OBJ_VER
  - NAMESPACE_OBJ_ID
  - NAMESPACE_OBJ_VER
  - OBJECT_VERSION_NUMBER

### 32.1.7 Undo Checkout Transformation Map

Use this API to undo check out the given transformation map.

Name** DME_PUB_XFORM_MAP.UndoCheckoutTransformationMap

**Signature:**

PROCEDURE UNDOCHECKOUTTRANSFORMATIONMAP
(P_API_VERSION IN VARCHAR2,
P_INIT_MSG_LIST IN VARCHAR2 DEFAULT CDR_PUB_DEF_CONSTANTS.G_FALSE,
P_COMMIT IN VARCHAR2 DEFAULT CDR_PUB_DEF_CONSTANTS.G_FALSE,
P_VALIDATION_LEVEL IN NUMBER DEFAULT CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
X_RETURN_STATUS OUT NOCOPY VARCHAR2,
X_MSG_COUNT OUT NOCOPY NUMBER,
X_MSG_DATA OUT NOCOPY VARCHAR2,
PIO_NAMING IN OUT NOCOPY CDR_NAMING_VERSION_OBJ_TYPE)

Parameters: This API has standard parameters (see “Standard Parameters” on page 2-5) as well as the following parameters:

- **PIO_NAMING** (Mandatory). This is a parameter of table type CDR_NAMING_VERSION_OBJ_TYPE that contains object attributes. Enter values to identify the transformation map.

  The following attributes are required: COMPANY_ID, OBJ_ID, OBJ_VER, NAMESPACE_OBJ_ID, NAMESPACE_OBJ_VER, OBJECT_VERSION_NUMBER.

### 32.1.8 Auto Map Tables

Use this API to create candidate auto Maps for table & Column transformation mappings. This operation accepts the target data model identifier object details as parameter for which auto maps has to be generated. The API creates the potential auto maps based on Oracle Name, Data Type & Length and Alias Match.

Name** DME_PUB_XFORM_MAP.AutoMapTables

**Signature:**

PROCEDURE AUTOMAPTABLES
CREATE AND MODIFY TRANSFORMATION MAPS

32.1.9 Accept Table Mappings

Use this API to create persistent Auto Maps by accepting the auto generated table and column transformation mappings created from dme_pub_xform_map.autoMapTables public API. User refers to DME_PUB_XFM_AUTOMAPS_V view to populate the attributes of input collection required as a parameter.

**Name**
DME_PUB_XFORM_MAP.AcceptTableAutoMappings

**Signature:**

PROCEDURE ACCEPTTABLEAUTOMAPPINGS

(P_API_VERSION IN VARCHAR2,
P_INIT_MSG_LIST IN VARCHAR2 DEFAULT CDR_PUB_DEF_CONSTANTS.G_FALSE,
P_COMMIT IN VARCHAR2 DEFAULT CDR_PUB_DEF_CONSTANTS.G_FALSE,
P_VALIDATION_LEVEL IN NUMBER DEFAULT CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
X_RETURN_STATUS OUT NOCOPY VARCHAR2,
X_MSG_COUNT OUT NOCOPY NUMBER,
X_MSG_DATA OUT NOCOPY VARCHAR2,
PI_TABLEAUTOMAPCOLL IN OUT NOCOPY DME_XFORM_AUTO_MAP_COLL
);

**Parameters:** This API has standard parameters (see "Standard Parameters" on page 2-5) as well as the following parameters:

- **PI_TABLEAUTOMAPCOLL** (Mandatory). This is a parameter of table type collection DME_XFORM_AUTO_MAP_COLL of DME_XFORM_AUTO_MAP_TYPE table type.

  All attributes are required including:
  - COMPANY_ID
  - MAP_ID
  - MAP_VER
  - MAP_ENTITY_ID
  - MAP_ENTITY_VER
  - CANDIDATE_FLAG_RC - with the possible values:
Create and Modify Transformation Maps

32.1.10 Auto Map Columns

Use this API to create candidate auto Maps for Column transformation mappings. This operation accepts the target table object identifier details as parameter for which auto maps has to be generated. The API creates the potential auto maps based on Oracle Name, Data Type & Length and Alias Match.

**Name**  DME_PUB_XFORM_MAP.AutoMapColumns

**Signature:**

```sql
PROCEDURE AUTOMAPCOLUMNS
    ( P_API_VERSION IN VARCHAR2,
      P_INIT_MSG_LIST IN VARCHAR2 DEFAULT CDR_PUB_DEF_CONSTANTS.G_FALSE,
      P_COMMIT IN VARCHAR2 DEFAULT CDR_PUB_DEF_CONSTANTS.G_FALSE,
      P_VALIDATION_LEVEL IN NUMBER DEFAULT CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
      X_RETURN_STATUS OUT NOCOPY VARCHAR2,
      X_MSG_COUNT OUT NOCOPY NUMBER,
      X_MSG_DATA OUT NOCOPY VARCHAR2,
      PI_DATAENTITYID IN CDR_NAMING_VERSIONS.OBJ_ID%TYPE,
      PI_DATAENTITYVER IN CDR_NAMING_VERSIONS.OBJ_VER%TYPE );
```

**Parameters:** This API has standard parameters (see “Standard Parameters” on page 2-5) as well as the following parameters:

- **PI_DATAENTITYID** (Mandatory). Enter the object Id of the target table on which auto map has to be performed.
- **PI_DATAENTITYVER** (Mandatory). Enter the object Version of the target table.

32.1.11 Accept Column Mappings

Use this API to create persistent Auto Maps by accepting the auto generated column transformation mappings.

**Name**  DME_PUB_XFORM_MAP.AcceptColumnAutoMappings

**Signature:**

```sql
PROCEDURE ACCEPTCOLUMNAUTOMAPPINGS
    ( P_API_VERSION IN VARCHAR2,
      P_INIT_MSG_LIST IN VARCHAR2 DEFAULT CDR_PUB_DEF_CONSTANTS.G_FALSE,
      P_COMMIT IN VARCHAR2 DEFAULT CDR_PUB_DEF_CONSTANTS.G_FALSE,
      P_VALIDATION_LEVEL IN NUMBER DEFAULT CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
      X_RETURN_STATUS OUT NOCOPY VARCHAR2,
      X_MSG_COUNT OUT NOCOPY NUMBER,
      X_MSG_DATA OUT NOCOPY VARCHAR2,
      PI_COLUMNAUTOMAPCOLL IN OUT NOCOPY DME_XFORM_AUTO_MAP_COLL );
```

**Parameters:** This API has standard parameters (see “Standard Parameters” on page 2-5) as well as the following parameters:
- **PI_COLUMNAUTOMAPCOLL** (Mandatory). This is a parameter of table type collection DME_XFORMAUTO_MAP_COLL of DME_XFORMAUTO_MAP_TYPE table type. All attributes are required including:
  - COMPANY_ID
  - MAP_ID
  - MAP_VER
  - MAP_ENTITY_ID
  - MAP_ENTITY_VER
  - CANDIDATE_FLAG_RC - with the possible values:
    * $YESNO$YES
    * $YESNO$NO - use this value for auto map suggestions which are applicable to save as real maps.

### 32.1.12 Upgrade Transformation Map

Use this API to upgrade the maps defined at model level.

**Name**  DME_PUB_XFORM_MAP.UpgradeXformMaps

**Signature:**

```sql
PROCEDURE UPGRADEXFORMMAPS
(P_API_VERSION IN VARCHAR2,
P_INIT_MSG_LIST IN VARCHAR2 DEFAULT CDR_PUB_DEF_CONSTANTS.G_FALSE,
P_COMMIT IN VARCHAR2 DEFAULT CDR_PUB_DEF_CONSTANTS.G_FALSE,
P_VALIDATION_LEVEL IN NUMBER DEFAULT CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
X_RETURN_STATUS OUT NOCOPY VARCHAR2,
X_MSG_COUNT OUT NOCOPY NUMBER,
X_MSG_DATA OUT NOCOPY VARCHAR2
);
```

**Parameters:**  This API has standard parameters (see "Standard Parameters" on page 2-5) as well as the following parameters:

- **PI_XFORMMAPCOMPID** (Mandatory). Enter the COMPANY_ID for the transformation map to be upgraded.
- **PI_XFORMMAPOBJID** (Mandatory). Enter the OBJ_ID for the transformation map to be upgraded.
- **PI_XFORMMAPOBJVER** (Mandatory). Enter the OBJ_VER for the transformation map to be upgraded.

### 32.1.13 Install Transformation Map

Use this API to install transformation mappings.

**Name**  DME_PUB_XFORM_MAP.InstallXform

**Signature:**

```sql
PROCEDURE INSTALLXFORM
(P_API_VERSION IN VARCHAR2,
P_INIT_MSG_LIST IN VARCHAR2 DEFAULT CDR_PUB_DEF_CONSTANTS.G_FALSE,
P_COMMIT IN VARCHAR2 DEFAULT CDR_PUB_DEF_CONSTANTS.G_FALSE,
P_VALIDATION_LEVEL IN NUMBER DEFAULT CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
X_RETURN_STATUS OUT NOCOPY VARCHAR2,
X_MSG_COUNT OUT NOCOPY NUMBER,
X_MSG_DATA OUT NOCOPY VARCHAR2
);
```
Create and Modify Transformation Maps

Transformations

32.13 Remove Transformation Map

Use this API to remove the transformation map at Model, Table or Column Level.

Name  DME_PUB_XFORM_MAP.RemoveTransformationMap

Signature:

PROCEDURE REMOVETRANSFORMATIONMAP
(P_API_VERSION IN VARCHAR2,
P_INIT_MSG_LIST IN VARCHAR2 DEFAULT CDR_PUB_DEF_CONSTANTS.G_FALSE,
P_COMMIT IN VARCHAR2 DEFAULT CDR_PUB_DEF_CONSTANTS.G_FALSE,
P_VALIDATION_LEVEL IN NUMBER DEFAULT CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
X_RETURN_STATUS OUT NOCOPY VARCHAR2,
X_MSG_COUNT OUT NOCOPY NUMBER,
X_MSG_DATA OUT NOCOPY VARCHAR2,
PIO_NAMING IN OUT NOCOPY CDR_NAMING_VERSION_OBJ_TYPE
);

Parameters:  This API has standard parameters (see “Standard Parameters” on page 2-5) as well as the following parameters:

■  PIO_NAMING (Mandatory). This is a parameter of table type CDR_NAMING_VERSION_OBJ_TYPE that contains object attributes. Enter values to identify the transformation map. The following attributes are required:

32.14 Remove Transformation Map

Use this API to remove the transformation map at Model, Table or Column Level.

Name  DME_PUB_XFORM_MAP.RemoveTransformationMap

Signature:

PROCEDURE REMOVETRANSFORMATIONMAP
(P_API_VERSION IN VARCHAR2,
P_INIT_MSG_LIST IN VARCHAR2 DEFAULT CDR_PUB_DEF_CONSTANTS.G_FALSE,
P_COMMIT IN VARCHAR2 DEFAULT CDR_PUB_DEF_CONSTANTS.G_FALSE,
P_VALIDATION_LEVEL IN NUMBER DEFAULT CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
X_RETURN_STATUS OUT NOCOPY VARCHAR2,
X_MSG_COUNT OUT NOCOPY NUMBER,
X_MSG_DATA OUT NOCOPY VARCHAR2,
PIO_NAMING IN OUT NOCOPY CDR_NAMING_VERSION_OBJ_TYPE
);

Parameters:  This API has standard parameters (see “Standard Parameters” on page 2-5) as well as the following parameters:

■  PIO_NAMING (Mandatory). This is a parameter of table type CDR_NAMING_VERSION_OBJ_TYPE that contains object attributes. Enter values to identify the transformation map. The following attributes are required:
32.1.15 Validate Transformation Maps

Use this API to validate the transformation maps defined at model and table level. If a transformation map is invalid, the status of transformation map is invalidated to 'Invalid', description field is updated with validation errors and red icon is shown on UI corresponding to invalid transformation maps.

Name  DME_PUB_XFORM_MAP.ValidateXform

Signature:

PROCEDURE VALIDATEXFORM
(P_API_VERSION IN VARCHAR2,
 P_INIT_MSG_LIST IN VARCHAR2 DEFAULT CDR_PUB_DEF_CONSTANTS.G_FALSE,
 P_COMMIT IN VARCHAR2 DEFAULT CDR_PUB_DEF_CONSTANTS.G_FALSE,
 P_VALIDATION_LEVEL IN NUMBER DEFAULT CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
 X_RETURN_STATUS OUT NOCOPY VARCHAR2,
 X_MSG_COUNT OUT NOCOPY NUMBER,
 X_MSG_DATA OUT NOCOPY VARCHAR2,
 PIO_NAMING IN OUT NOCOPY CDR_NAMING_VERSION_OBJ_TYPE
);

Parameters:  This API has standard parameters (see "Standard Parameters" on page 2-5) as well as the following parameters:

- **PIO_NAMING** (Mandatory). This is a parameter of table type CDR_NAMING_VERSION_OBJ_TYPE that contains object attributes. Enter values to identify the transformation map. The following attributes are required:
  - COMPANY_ID
  - OBJ_ID
  - OBJ_VER

32.1.16 Update Validation Status

Use this API to modify the validation status.

Name  DME_PUB_XFORM_MAP.UpdateValStatus

Signature:

PROCEDURE UPDATEVALSTATUS
(P_API_VERSION IN VARCHAR2,
 P_INIT_MSG_LIST IN VARCHAR2 DEFAULT CDR_PUB_DEF_CONSTANTS.G_FALSE,
 P_COMMIT IN VARCHAR2 DEFAULT CDR_PUB_DEF_CONSTANTS.G_FALSE,
 P_VALIDATION_LEVEL IN NUMBER DEFAULT CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
 X_RETURN_STATUS OUT NOCOPY VARCHAR2,
 X_MSG_COUNT OUT NOCOPY NUMBER,
 X_MSG_DATA OUT NOCOPY VARCHAR2,
 PIO_NAMING IN OUT NOCOPY CDR_NAMING_VERSION_OBJ_TYPE
);

- COMPANY_ID
- OBJ_ID
- OBJ_VER

32.1.16 Update Validation Status

Use this API to modify the validation status.
Parameters: This API has standard parameters (see "Standard Parameters" on page 2-5) as well as the following parameters:

- **PIO_NAMING** (Mandatory). This is a parameter of table type CDR_NAMING_VERSION_OBJ_TYPE that contains object attributes. All attributes are required including:
  - COMPANY_ID
  - OBJ_ID
  - OBJ_VER
  - NAMESPACE_OBJ_ID
  - NAMESPACE_OBJ_VER
  - OBJECT_TYPE_RC
  - CHECKED_OUT_FLAG_RC
  - VALIDATION_STATUS_RC

The new validation status should be assigned to field VALIDATION_STATUS_RC. The possible valid values for validation status are:

- $SYSVALDNSTEPS$DEVELOPMENT
- $SYSVALDNSTEPS$QUALITYCONTROL
- $SYSVALDNSTEPS$PRODUCTION
- $SYSVALDNSTEPS$RETIRED

32.1.17 Execute Transformation Map

Use this API to execute a Transformation Map.

**Name** DME_PUB_XFORM_MAP.ExecuteXform

**Signature:**

```sql
PROCEDURE EXECUTEXFORM
  (P_API_VERSION IN VARCHAR2,
   P_INIT_MSG_LIST IN VARCHAR2 DEFAULT CDR_PUB_DEF_CONSTANTS.G_FALSE,
   P_COMMIT IN VARCHAR2 DEFAULT CDR_PUB_DEF_CONSTANTS.G_FALSE,
   P_VALIDATION_LEVEL IN NUMBER DEFAULT CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
   X_RETURN_STATUS OUT NOCOPY VARCHAR2,
   X_MSG_COUNT OUT NOCOPY NUMBER,
   X_MSG_DATA OUT NOCOPY VARCHAR2,
   PI_MODELCOMPANYID IN CDR_NAMINGS.COMPANY_ID%TYPE,
   PI_MODELOBJID IN CDR_NAMINGS.OBJ_ID%TYPE,
   PI_MODELOBJVER IN CDR_NAMING_VERSIONS.OBJ_VER%TYPE,
   PI_SUBMISSION_TYPE IN VARCHAR2 DEFAULT '$SUBMISSTYPES$IMMEDIATE',
   PI_CONTEXT IN VARCHAR2 DEFAULT '$LIFECYCLE$DEV',
   PI_SCHED_START_TS IN DATE DEFAULT NULL,
   PI_SCHED_END_TS  IN DATE DEFAULT NULL,
   PI_SCHED_REPEAT_INTERVAL IN VARCHAR2 DEFAULT NULL,
   PI_SCHED_REPEAT_LIST IN VARCHAR2 DEFAULT NULL,
   PI_FORCE_EXECUTION_FLAG_RC IN VARCHAR2 DEFAULT '$YESNO$NO',
   PI_RUN_MODE_RC IN VARCHAR2,
   PI_TRIGGER_DOWNSTREAM IN VARCHAR2,
   PO_JOBID OUT NOCOPY CDR_JOBS.JOB_ID%TYPE )
);```
**Parameters:** This API has standard parameters (see "Standard Parameters" on page 2-5) as well as the following parameters:

- **PI_MODELCOMPANYID** (Mandatory). Company id of Target Data Model.
- **PI_MODELOBJID** (Mandatory). Object Id of Target Data Model.
- **PI_MODELOBJVER** (Mandatory). Object Version of Target Data Model
- **PI_SUBMISSION_TYPE**. Enter "$SUBMISSIONTYPES$IMMEDIATE or "$SUBMISSIONTYPES$SCHEDULED" or "$SUBMISSIONTYPES$DEFERRED".
- **PI_CONTEXT**. Lifecycle context values like "$LIFECYCLE$DEV", "$LIFECYCLE$QC" or "$LIFECYCLE$PROD".
- **PI_SCHED_START_TS**. Enter start time if submission is scheduled/deferred.
- **PI_SCHED_END_TS**. Enter end time if submission is scheduled.
- **PI_SCHED_REPEAT_INTERVAL**. Enter interval time if submission is scheduled.
- **PI_SCHED_REPEAT_LIST**. Enter Unit like Hour(s), Day(s), Week(s).
- **PI_FORCE_EXECUTION_FLAG_RC**. Enter "$YESNO$NO or "$YESNO$YES".
- **PI_RUN_MODE_RC**. Enter "$RUNMODES$FULL" or "$RUNMODES$INCREMENT".
- **PI_TRIGGER_DOWNSTREAM**. Enter "$YESNO$YES" or "$YESNO$NO".

### 32.1.18 Create Staging Table

Use this API to create a new Staging Table.

**Name**  DME_PUB_XFORM_MAP.CreateStagingTable

**Signature:**

```sql
PROCEDURE CREATESTAGINGTABLE
  (P_API_VERSION IN VARCHAR2,
   P_INIT_MSG_LIST IN VARCHAR2 DEFAULT CDR_PUB_DEF_CONSTANTS.G_FALSE,
   P_COMMIT IN VARCHAR2 DEFAULT CDR_PUB_DEF_CONSTANTS.G_FALSE,
   P_VALIDATION_LEVEL IN NUMBER DEFAULT CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
   X_RETURN_STATUS OUT NOCOPY VARCHAR2,
   X_MSG_COUNT OUT NOCOPY NUMBER,
   X_MSG_DATA OUT NOCOPY VARCHAR2,
   PI_XFORMMAPCOLL IN OUT NOCOPY DME_XFORM_MAP_EX_COLL,
   PI_EXISTINGSECMODELOBJ IN CDR_BASE_OBJ_TYPE
 );
```

**Parameters:** This API has standard parameters (see "Standard Parameters" on page 2-5) as well as the following parameters:

- **PI_XFORMMAPCOLL** (Mandatory). This is a parameter of collection type DME_XFORM_MAP_EX_COLL and DME_XFORM_MAP_EX_COLL is table of DME_XFORM_MAP_EX_OBJ_TYPE object type.
  The collection is populated with both table and column mappings with respective source and target entities. Staging table details are set at table level. Staging table can be created from one or more source tables.
  The attributes required for this API from DME_XFORM_MAP_EX_OBJ_TYPE are:
  - NAMING: Table type CDR_NAMING_VERSION_OBJ_TYPE
- **COMPANY_ID** - Enter the Company Id.

- **OBJECT_TYPE_RC** - Enter the OBJECT_TYPE_RC as
  '$OBJTYPES$XFORMMAP'

- **NAME** - Enter the name of user defined staging table in table level mapping.

For Table & Column level mappings following additional attributes are required:

- **MAP_LEVEL** - Map_Level should be assigned as 'TABLE' for table maps and 'COLUMN' for column maps.

- **MAP_TYPE** - Map_Type should be passed as '$MAPTYPES$DEFAULT' for both table and column level.

- **XFORM_TYPE** - Map_Type should be passed as '$XFORMTYPES$DIRECT' for table maps.

- **DUP_NUM_FLAG** - This flag should be passed as '$YESNO$YES' to support duplicates.

- **OPERATION_TYPE** - The operation type should be passed as '$OPERCREATE'.

- **MAP_ENTITY_COLL** - This attribute is a collection of DME_MAP_ENTITY_OBJ_TYPE object type attributes.

For table level maps, this collection should be populated only with one or more Source tables from source data model from which user want to select the columns to use in staging table. For column level maps, this collection should be populated both with Source and Target column entity.

The attributes required for this API from DME_MAP_ENTITY_OBJ_TYPE are:

- **DATA_ENTITY_ID** - Enter Table Object Id for Table Level and or Column Object Id for Column Level Source entities. Leave blank for Column level target entity.

- **DATA_ENTITY_VER** - Enter Table Object Version for Table Level and or Column Object Version for Column Level Source entities. Leave blank for Column level target entity.

- **ALIAS** - Enter alias as Target Column Name for Column Level Target Map entity.

- **MAP_RELATION** - Map_relation should be passed as 'SOURCE' for source map entities and 'TARGET' for target map entities. The Table level mapping will only have 'SOURCE' map relation entities.

- **PRIMARY_KEY_FLAG** - This value should be passed as '$YESNO$YES' for the target column entities which are applicable to be primary key in staging table.

- **OPERATION_TYPE** - The operation type should be passed as '$OPERCREATE' for all source and target entities.

- **PI_EXISTINGSECMODELOBJ** (Mandatory). This is a parameter of CDR_BASE_OBJ_TYPE. Enter identifying attributes of the Target data model where you want to create the staging table. The following attributes are required:

  - **COMPANY_ID**
  - **OBJ_ID**
  - **OBJ_VER**
32.1.19 Validate Expression

Use this API to validate user defined map column or criteria expression.

**Name**  DME_PUB_XFORM_MAP.ValidateExpression

**Signature:**

PROCEDURE VALIDATEEXPRESSION
  (P_API_VERSION IN VARCHAR2,
   P_INIT_MSG_LIST IN VARCHAR2 DEFAULT CDR_PUB_DEF_CONSTANTS.G_FALSE,
   P_COMMIT IN VARCHAR2 DEFAULT CDR_PUB_DEF_CONSTANTS.G_FALSE,
   P_VALIDATION_LEVEL IN NUMBER DEFAULT CDR_PUB_DEF_CONSTANTS.G_VALID_LEVEL_FULL,
   X_RETURN_STATUS OUT NOCOPY VARCHAR2,
   X_MSG_COUNT OUT NOCOPY NUMBER,
   X_MSG_DATA OUT NOCOPY VARCHAR2,
   PI_OBJID IN CDR_NAMINGS.OBJ_ID%TYPE,
   PI_OBJVER IN CDR_NAMING_VERSIONS.OBJ_VER%TYPE,
   PI_EXPROBJ IN DME_XFORM_EXPR_OBJ_TYPE
  );

**Parameters:**  This API has standard parameters (see "Standard Parameters" on page 2-5) as well as the following parameters:

- **PI_OBJID** (Mandatory). Enter the object Id of the model transformation map in which expression is applied for one of the map entity. In case of Create operations, map will not exist in DB. Pass the table definition id from which column has been selected into the expression. Pass NULL, in case, no column is used in expression text.

- **PI_OBJVER** (Mandatory). Enter the object Version of the model transformation map or table definition from which column is used in expression. Pass NULL, in case there is no column used in expression.

- **PI_EXPROBJ** (Mandatory). This parameter is a object type of DME_XFORM_EXPR_OBJ_TYPE. Refer to the following section Chapter 28.2, "Create or Modify an Expression," for more details.
This part of the Oracle Life Sciences Data Hub (Oracle LSH) API guide contains the public views for Oracle Health Sciences Data Management Workbench (DMW) APIs. Part V contains the following chapters:

- Section 33, "Public Views"
This section includes the public views for the following APIs:

- Section 33.1, "Code Lists Views"
- Section 33.2, "Validation Checks Views"
- Section 33.3, "Transformation Views"

### 33.1 Code Lists Views

This section contains the following public APIs views:

- Section 33.1.1, "DME_PUB_CODELIST_V"
- Section 33.1.2, "DME_PUB_CODELIST_VALUES_V"

#### 33.1.1 DME_PUB_CODELIST_V

**Purpose**  Use this view to see code list data.

#### 33.1.2 DME_PUB_CODELIST_VALUES_V

**Purpose**  Use this view to see code list values data.

### 33.2 Validation Checks Views

This section contains the following public APIs views:

- Section 33.2.1, "DME_PUB_VC_BATCHES_V"
- Section 33.2.2, "DME_PUB_VC_DETAILS_V"

#### 33.2.1 DME_PUB_VC_BATCHES_V

**Purpose**  Use this view to see details about Validation Checks Batches. The following query lists out the Validation Checks Batches available for a given Data model, resulting details about the batch:

```sql
Select * from DME_MT_VC_BATCHES_V
where SRC_MODEL_COMPANY_ID = <company_id> and
SRC_MODEL_OBJ_ID = <obj> and
SRC_MODEL_OBJ_VER = <ver>;
```
The following parameters are required for operations related to a batch, e.g. update, checkin, uncheck, checkout, install etc.

BATCH_MAP_COMPANY_ID
BATCH_MAP_OBJ_ID
BATCH_MAP_OBJ_VER
BATCH_MAP_NS_OBJ_ID
BATCH_MAP_NS_OBJ_VER
BATCH_MAP_OBJECT_VER_NUM

33.2.2 DME_PUB_VCDETAILS_V

**Purpose**  Use this view to see details about Validation Checks. The following query lists out the Validation Checks available for a given Validation Check Batch.

Select * from DME_MT_VC_DETAILS_V
where vc_company_id= <vc_batch_map_company_id>
and VC_BATCH_MAP_OBJ_ID = <vc_batch_map_id>
and <vc_batch_map_version>
between VC_MAP_NS_START_OBJ_VER and VC_MAP_NS_END_OBJ_VER;

33.3 Transformation Views

This section contains the following public APIs views:

- Section 33.3.1, "DME_PUB_DF_XFORM_MAP_V"
- Section 33.3.2, "DME_PUB_DF_MAP_ENTITY_V"
- Section 33.3.3, "DME_PUB_XFM_SOURCE_TABLES_V"
- Section 33.3.4, "DME_PUB_XFM_SOURCE_COLUMNS_V"
- Section 33.3.5, "DME_PUB_XFM_TARGET_TABLES_V"
- Section 33.3.6, "DME_PUB_XFM_TARGET_COLUMNS_V"
- Section 33.3.7, "DME_PUB_XFM_AUTOMAPS_V"
- Section 33.3.8, "DME_PUB_XFM_COL_AUTOMAPS_V"
- Section 33.3.9, "DME_PUB_XFM_CUSTOMPROGRAMS_V"
- Section 33.3.10, "DME_PUB_XFM_EXPRSTDFUNC_V"
- Section 33.3.11, "DME_PUB_XFM_EXPRSTATICPKGS_V"

33.3.1 DME_PUB_DF_XFORM_MAP_V

**Purpose**  Use this view to see transformation map details.

33.3.2 DME_PUB_DF_MAP_ENTITY_V

**Purpose**  Use this view to see map entity details.

33.3.3 DME_PUB_XFM_SOURCE_TABLES_V

**Purpose**  Use this view to see source database tables involved in table mappings.
33.3.4 **DME_PUB_XFM_SOURCE_COLUMNS_V**

**Purpose**  Use this view to see details of columns from the source database tables, which are mapped to columns in target database table.

33.3.5 **DME_PUB_XFM_TARGET_TABLES_V**

**Purpose**  Use this view to see details for target database tables involved in table mappings.

33.3.6 **DME_PUB_XFM_TARGET_COLUMNS_V**

**Purpose**  Use this view to see details of the columns from the target database tables, to which one or more source table are mapped.

33.3.7 **DME_PUB_XFM_AUTOMAPS_V**

**Purpose**  Use this view to see details of the available auto maps at table level.

33.3.8 **DME_PUB_XFM_COL_AUTOMAPS_V**

**Purpose**  Use this view to see details of available auto maps at column level.

33.3.9 **DME_PUB_XFM_CUSTOM_PROGRAMS_V**

**Purpose**  Use this view to see the details of the available custom programs.

33.3.10 **DME_PUB_XFM_EXPR_STDFUNC_V**

**Purpose**  Use this view to see supported standard Oracle Scalar functions and their respective function return types. You can refer to these functions while creating the expressions.

33.3.11 **DME_PUB_XFM_EXPR_STATIC_PKGS_V**

**Purpose**  Use this view to see the user defined LSH stored functions Scalar functions and their respective details. You can check available user-defined functions from this view and can use these functions while dealing with expressions.