Best Practices for SAP HANA Smart Data Integration and SAP HANA Smart Data Quality
# Typographic Conventions

<table>
<thead>
<tr>
<th>Type Style</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><em>Example</em></td>
<td>Words or characters quoted from the screen. These include field names, screen titles, pushbuttons labels, menu names, menu paths, and menu options. Textual cross-references to other documents.</td>
</tr>
<tr>
<td><em>Example</em></td>
<td>Emphasized words or expressions.</td>
</tr>
<tr>
<td><strong>EXAMPLE</strong></td>
<td>Technical names of system objects. These include report names, program names, transaction codes, table names, and key concepts of a programming language when they are surrounded by body text, for example, <code>SELECT</code> and <code>INCLUDE</code>.</td>
</tr>
<tr>
<td><em>Example</em></td>
<td>Output on the screen. This includes file and directory names and their paths, messages, names of variables and parameters, source text, and names of installation, upgrade and database tools.</td>
</tr>
<tr>
<td><em>Example</em></td>
<td>Exact user entry. These are words or characters that you enter in the system exactly as they appear in the documentation.</td>
</tr>
<tr>
<td><code>&lt;Example&gt;</code></td>
<td>Variable user entry. Angle brackets indicate that you replace these words and characters with appropriate entries to make entries in the system.</td>
</tr>
<tr>
<td><strong>EXAMPLE</strong></td>
<td>Keys on the keyboard, for example, <code>F2</code> or <code>ENTER</code>.</td>
</tr>
</tbody>
</table>
## Document History

<table>
<thead>
<tr>
<th>Version</th>
<th>Date</th>
<th>Change</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.0</td>
<td>2016-06-30</td>
<td>Include naming conventions, implementation guidelines, and implementation scenario.</td>
</tr>
<tr>
<td>2.0</td>
<td>2016-09-30</td>
<td>Add integration landscape and tuning recommendations.</td>
</tr>
<tr>
<td>2.1</td>
<td>2016-12-15</td>
<td>Replace source partitioning with task partitioning.</td>
</tr>
<tr>
<td>2.2</td>
<td>2017-01-03</td>
<td>Correct minor typographical errors and wording.</td>
</tr>
<tr>
<td>2.3</td>
<td>2017-02-01</td>
<td>Correct minor link errors.</td>
</tr>
<tr>
<td>2.4</td>
<td>2017-03-22</td>
<td>Update transportable SAP Core Data Services artifacts.</td>
</tr>
</tbody>
</table>
# Contents

1 Introduction ................................................................................................................................... 5

2 Naming Conventions..................................................................................................................... 6
   2.1 Suggested Naming Conventions ............................................................................................... 6
   2.1.1 Subject Area .......................................................................................................................... 6
   2.1.2 Additional Naming Conventions .......................................................................................... 6
   2.2 Naming Convention Details ...................................................................................................... 7
   2.2.1 Package .................................................................................................................................. 7
   2.2.2 Remote Source ....................................................................................................................... 7
   2.2.3 Virtual Table ........................................................................................................................... 7
   2.2.4 Replication Task .................................................................................................................... 7
   2.2.5 Flowgraph ............................................................................................................................. 8
   2.2.6 Transformation Node ............................................................................................................ 9

3 Deployment Guidelines ............................................................................................................... 11
   3.1 Data Provisioning Agent Deployment ......................................................................................... 11
   3.1.1 Data Source Location ........................................................................................................... 11
   3.1.2 Source System Operating System ..................................................................................... 11
   3.1.3 Data Replication Type .......................................................................................................... 12

4 Implementation Guidelines ........................................................................................................ 13
   4.1 Editor .......................................................................................................................................... 13
   4.2 Replication Tasks ...................................................................................................................... 13

5 Implementation Scenarios ......................................................................................................... 15
   5.1 Preparation ................................................................................................................................ 15
   5.2 Transportable Objects ............................................................................................................... 16
   5.3 Scenario ...................................................................................................................................... 16

6 Tuning Guidelines ....................................................................................................................... 20
   6.1 Initial Load/Batch Replication .................................................................................................... 20
   6.1.1 Sample Data Provisioning Agent Tuning ............................................................................. 20
   6.1.2 Review Performance and Identify Bottlenecks ................................................................... 20
   6.1.3 Task Partitioning .................................................................................................................. 21
   6.1.4 Target Table Partitioning ...................................................................................................... 22
   6.2 Real-time Replication ................................................................................................................ 23
   6.2.1 Measuring Latency ............................................................................................................... 23
   6.2.2 Throughput ........................................................................................................................... 24
   6.2.3 Resource Usage .................................................................................................................... 24
   6.2.4 Tuning Customization .......................................................................................................... 25
1 Introduction

In most implementation projects, several project members have different roles and collaborate to accomplish a variety of tasks. This document provides general guidelines that an implementation project team can follow to use SAP HANA smart data integration and SAP HANA smart data quality successfully.

Best practices introduced in this document include the following:

- Suggested naming conventions for commonly used information objects in SAP HANA smart data integration and SAP HANA smart data quality

- Required authorizations and high-level implementation workflows for the following:
  - Creating users for SAP HANA smart data integration and SAP HANA smart data quality
  - Creating and sharing remote sources
  - Creating and sharing projects
  - Creating replication tasks and flowgraphs
  - Transporting replication tasks and flowgraphs between landscapes
    For example, transporting a flowgraph from the development landscape to the production landscape.

Assumptions and General Notes

- Each individual in the implementation product team is familiar with the existing product documentation. Detailed information for specific tasks (such as creating remote sources, creating flowgraphs, and so on) is available in the product documentation available on the Help Portal: http://help.sap.com/hana_options_eim

- Unless specified, all concepts are generic and independent of specific product versions.

- Unless specified, all tasks can be performed in the SAP HANA Web-based Development Workbench.
2 Naming Conventions

By consistently naming objects required for SAP HANA smart data integration and SAP HANA smart data quality, such as flowgraphs and replication tasks, you can more easily identify each object’s function and type.

2.1 Suggested Naming Conventions

<table>
<thead>
<tr>
<th>Object Type</th>
<th>Suggested Naming Convention</th>
</tr>
</thead>
<tbody>
<tr>
<td>Package</td>
<td>PKG_&lt;SubjectArea&gt;</td>
</tr>
<tr>
<td>Remote Source</td>
<td>RS_&lt;SourceType&gt;</td>
</tr>
<tr>
<td>Replication Task</td>
<td>REP_&lt;ReplicationBehavior&gt;<em>&lt;SubjectArea&gt;</em>&lt;action&gt;</td>
</tr>
<tr>
<td>Flowgraph</td>
<td>FG_&lt;RuntimeBehavior&gt;<em>&lt;SubjectArea&gt;</em>&lt;action&gt;</td>
</tr>
<tr>
<td>Transformation Node</td>
<td>&lt;NodeType&gt;<em>&lt;SubjectArea&gt;</em>&lt;description&gt;</td>
</tr>
</tbody>
</table>

i Note
For information about the maximum lengths of object names, see the SAP HANA platform documentation on the Help Portal: http://help.sap.com/hana_platform

2.1.1 Subject Area

The subject area portion of the object naming convention indicates the area of the application that the object is involved with. For example, for a business warehouse phase-1 conversion project, the subject area portion of the naming convention might be BW-PHASE1-CONV.

2.1.2 Additional Naming Conventions

You can use naming standards for other components that may be involved in flowgraphs or replication tasks.

<table>
<thead>
<tr>
<th>Object Type</th>
<th>Suggested Naming Convention</th>
</tr>
</thead>
<tbody>
<tr>
<td>Expression variables</td>
<td>V_EX_&lt;action&gt;</td>
</tr>
<tr>
<td>Scalar parameter variables</td>
<td>V_SP_&lt;action&gt;</td>
</tr>
</tbody>
</table>
2.2 Naming Convention Details

2.2.1 Package

A package is an object used to group related SAP HANA information objects in a structured way. Packages group all of the information models and make it easier to transport models in an efficient manner.

Recommendation
Prefix package names with PKG_ and include the subject area:
PKG_<SubjectArea>

2.2.2 Remote Source

SAP HANA remote source objects point to the remote systems or other external data sources containing the data that is replicated into or accessed by SAP HANA.

Recommendation
Prefix remote source names with RS_ and include the source type:
RS_<SourceType>

2.2.3 Virtual Table

SAP HANA virtual tables are pointers to remote tables. They look and feel like local tables and have all metadata stored locally in SAP HANA, but all data is remote - no records are stored in SAP HANA.

Recommendation
Prefix virtual table names with VT_ and include the table name:
VT_<TableName>

2.2.4 Replication Task

Replication tasks replicate data from various objects in a remote source to tables within SAP HANA and are created in the Replication Editor in the SAP HANA Web-based Development Workbench.
To replicate data from objects in a remote source into SAP HANA tables, you must configure the replication process by creating an .hdbreptask file, which opens a file specific to the Replication Editor.
Recommendation
Prefix replication task names with REP_ and include the replication behavior and subject area:
REP_<ReplicationBehavior>_<SubjectArea>_<action>

Suggested Naming Conventions for Replication Behavior

<table>
<thead>
<tr>
<th>Replication Behavior</th>
<th>Suggested Naming Convention</th>
</tr>
</thead>
<tbody>
<tr>
<td>Initial load only</td>
<td>INT</td>
</tr>
<tr>
<td>Initial + Realtime</td>
<td>WITHCDC</td>
</tr>
<tr>
<td>Realtime</td>
<td>CDC</td>
</tr>
<tr>
<td>No data transfer</td>
<td>STRC</td>
</tr>
<tr>
<td>Initial + Realtime with structure</td>
<td>WITHSTRC</td>
</tr>
<tr>
<td>Realtime only with structure</td>
<td>CDCSTRC</td>
</tr>
</tbody>
</table>

Examples
- Replicate tables with initial load only from global finance management system source Oracle10g.
  REP_INT_GFMS_ORA10g
- Replicate tables with CDC applied from Global Finance Management system source Oracle10g, and CUSTLOC TABLE with custom columns defined in the projection.
  REP_WITHCDC_GFMS_ORA-CUSTLOC-CUSTOM

2.2.5 Flowgraph

In SAP HANA smart data integration and SAP HANA smart data quality, flowgraphs model data flows that may include the following:
- Tables, views, and procedures from the SAP HANA catalog
- Transformations such as projections, filters, unions, and joins
- Functions from Application Framework Modeling (AFM). For example, AFL, R, and so on.
- Attribute views and calculation views, and so on.
- Runtime behaviors

Recommendation
Prefix flowgraph names with FG_ and include the runtime behavior and subject area:
FG_<RuntimeBehavior>_<SubjectArea>_<action>
### Suggested Naming Conventions for Runtime Behavior

<table>
<thead>
<tr>
<th>Runtime Behavior</th>
<th>Suggested Naming Convention</th>
</tr>
</thead>
<tbody>
<tr>
<td>Batch Task</td>
<td>BATCH</td>
</tr>
<tr>
<td>Procedure</td>
<td>PROC</td>
</tr>
<tr>
<td>Realtime Task</td>
<td>RT</td>
</tr>
<tr>
<td>Transactional Task</td>
<td>TRANS</td>
</tr>
</tbody>
</table>

**Examples**

- Extract customer data in batch mode.
  
  `FG_BATCH_CUSTOMER_EXTRACTFILE`

- Apply a cleanse transformation to customer data in realtime mode.
  
  `FG_RT_CUSTOMER_TRANSFORMCLEANSE`

- Generate customer data for transactional mode in file format.
  
  `FG_TRANS_CUSTOMER_OUTPUTFILE`

- Extract customer from Oracle using a procedure.

  `FG_PROC_CUSTOMER_EXTRATORA`

### 2.2.6 Transformation Node

In SAP HANA smart data integration and SAP HANA smart data quality, flowgraphs consist of input, output, and configurable nodes that perform data transformations. Transformation node names may contain only letters, numbers, and underscores, and must be unique within a flowgraph.

### Suggested Transformation Node Abbreviations

<table>
<thead>
<tr>
<th>Node Type</th>
<th>Abbreviation</th>
<th>Node Type</th>
<th>Abbreviation</th>
</tr>
</thead>
<tbody>
<tr>
<td>INPUT TYPE</td>
<td>IN_TT</td>
<td>DATE GENERATION</td>
<td>DG</td>
</tr>
<tr>
<td>OUTPUT TYPE</td>
<td>OUT_TT</td>
<td>ROW GENERATION</td>
<td>RG</td>
</tr>
<tr>
<td>DATA SOURCE</td>
<td>DS</td>
<td>CASE</td>
<td>CASE</td>
</tr>
<tr>
<td>DATA SINK</td>
<td>DSK</td>
<td>PIVOT</td>
<td>PVT</td>
</tr>
<tr>
<td>TEMPLATE TABLE</td>
<td>TMT</td>
<td>UNPIVOT</td>
<td>UNPVT</td>
</tr>
<tr>
<td>TEMPLATE FILE</td>
<td>TF</td>
<td>LOOKUP</td>
<td>LKP</td>
</tr>
<tr>
<td>AGGREGATION</td>
<td>AG</td>
<td>CLEANSE</td>
<td>CLNS</td>
</tr>
<tr>
<td>FILTER</td>
<td>FL</td>
<td>GEOCODE</td>
<td>GEO</td>
</tr>
<tr>
<td>Node Type</td>
<td>Abbreviation</td>
<td>Node Type</td>
<td>Abbreviation</td>
</tr>
<tr>
<td>------------------------</td>
<td>--------------</td>
<td>--------------------------</td>
<td>--------------</td>
</tr>
<tr>
<td>JOIN</td>
<td>JN</td>
<td>MATCH</td>
<td>MAT</td>
</tr>
<tr>
<td>SORT</td>
<td>SO</td>
<td>TABLE COMPARISON</td>
<td>TC</td>
</tr>
<tr>
<td>UNION</td>
<td>UN</td>
<td>MAP OPERATION</td>
<td>MO</td>
</tr>
<tr>
<td>PROCEDURE</td>
<td>PROC</td>
<td>HISTORY PRESERVING</td>
<td>HP</td>
</tr>
<tr>
<td>Rscript</td>
<td>SC</td>
<td>HIERARCHICAL</td>
<td>HRCL</td>
</tr>
<tr>
<td>AFL function</td>
<td>FUNC</td>
<td>DATA MASKING</td>
<td>DM</td>
</tr>
</tbody>
</table>
3  Deployment Guidelines

3.1  Data Provisioning Agent Deployment

This section provides basic guidelines for deploying the Data Provisioning Agent in your landscape.

3.1.1  Data Source Location

We generally recommend that you install the Data Provisioning Agent close to the data sources. If you have data sources scattered across multiple geographical locations separated by distance and network latency, you can deploy multiple Data Provisioning Agents. Install at least one in each geographical location.

Generally speaking, the best performance is achieved when the agent is directly installed on the same server as the data source. However, various operational and IT policy reasons may prevent you from installing the agent directly on the data source servers. In these situations, we recommend that you install the Data Provisioning Agent on a supported VM close to the source.

When there is a firewall between the Data Provisioning Agent and the Data Provisioning Server in SAP HANA, the connection is automatically configured to use HTTP/S mode. When using HTTP/S, gzip compression is used by default.

When the Data Provisioning Agent connects to the SAP HANA server over TCP/IP, compression is not enabled by default, because the network latency is assumed to be negligible due to geographic proximity. If the TCP/IP connection introduces significant network latency, configure compression using the Data Provisioning Agent Configuration tool.

3.1.2  Source System Operating System

The Data Provisioning Agent can be installed on various versions of Windows and Linux. The operating system required for the Data Provisioning Agent may depend on the operating system of the data source system, when applicable.

For the latest complete information about operating system support for the Data Provisioning Agent and data sources, refer to the Product Availability Matrix (PAM).
3.1.3 Data Replication Type

If you plan to perform both batch and real-time replication from various source systems, we recommend that you divide batch and real-time tasks, assigning one agent for real-time replication and another for batch.
4 Implementation Guidelines

4.1 Editor

In general, for SAP HANA smart data integration and SAP HANA smart data quality tasks, consider using the SAP HANA Web-based Development Workbench instead of SAP HANA Studio.

4.2 Replication Tasks

The Replication Editor in the SAP HANA Web-based Development Workbench can be used to create tasks for replication of single or multiple remote source objects to SAP HANA tables.

- When you configure and save a replication task, additional objects are generated: a stored procedure, a remote subscription, one or more virtual tables, and target tables.
- When you execute a replication task, the stored procedure is called, and the initial load is performed. If real-time replication is enabled, a remote subscription is created and queued, and subsequent source changes are automatically distributed.

Guidelines for Mass Replication

Individual replication tasks may contain any number of sources and target tables. Before you use a replication task to replicate many remote source objects to SAP HANA tables, consider the following guidelines for your replication strategy:

- Follow a consistent naming scheme, and use separate schemas for target and virtual tables. For example, prefix virtual tables with VT_.
- Avoid modifying an active replication task unless resetting the task is acceptable. When you modify a replication task that has already been executed, the replication task must be saved and reactivated. This drops and recreates the associated stored procedure, remote subscription, virtual tables, and target tables.
- Avoid grouping source objects that require changes to the SAP HANA target structure into a single replication task with source objects that do not require structure changes. For source objects that require frequent structure updates, consider using separate individual replication tasks.
- Consider grouping wide or large source objects into a single replication task, so that initial loads requiring high memory consumption is performed sequentially.
When you group multiple source objects into a single replication task, the initial load for each source object is performed sequentially, while later real-time replication is performed in parallel.

If you use separate replication tasks for large source objects, the initial load for each source object may be performed in parallel, potentially causing an out of memory condition depending on the resources available on the SAP HANA system.
5  Implementation Scenarios

5.1  Preparation

A set of SAP HANA users are created to perform various tasks when implementing and using SAP HANA smart data integration and SAP HANA smart data quality.

<table>
<thead>
<tr>
<th>User</th>
<th>Responsibilities</th>
</tr>
</thead>
<tbody>
<tr>
<td>SDI_ADMIN</td>
<td>SAP HANA smart data integration administrator</td>
</tr>
<tr>
<td></td>
<td>• Creates a common project/package</td>
</tr>
<tr>
<td></td>
<td>• Creates remote sources</td>
</tr>
<tr>
<td></td>
<td>• Shares remote sources with project team members</td>
</tr>
<tr>
<td></td>
<td>• Shares replication tasks and flowgraphs</td>
</tr>
<tr>
<td></td>
<td>• Transports flowgraphs and replication tasks from one SAP HANA system to another</td>
</tr>
<tr>
<td></td>
<td>For example, from Development to Test to Production</td>
</tr>
<tr>
<td>SDI_DEV</td>
<td>IT or ETL developer</td>
</tr>
<tr>
<td></td>
<td>• Creates replication tasks and flowgraphs</td>
</tr>
<tr>
<td></td>
<td>• Executes replication tasks and flowgraphs in development</td>
</tr>
<tr>
<td></td>
<td>• Monitors flowgraph and replication task execution in development</td>
</tr>
<tr>
<td></td>
<td>• Shares replication tasks and flowgraphs with project team members</td>
</tr>
<tr>
<td>SDI_TRANSPORT</td>
<td>Object transport administration user</td>
</tr>
<tr>
<td></td>
<td>• Exports replication tasks and flowgraphs as a delivery unit</td>
</tr>
<tr>
<td></td>
<td>• Imports delivery units containing replication tasks and flowgraphs into a target</td>
</tr>
<tr>
<td></td>
<td>system</td>
</tr>
<tr>
<td>SDI_EXEC</td>
<td>Task execution user</td>
</tr>
<tr>
<td></td>
<td>• Executes flowgraphs and replication tasks in production</td>
</tr>
<tr>
<td></td>
<td>• Monitors flowgraph and replication task execution</td>
</tr>
</tbody>
</table>

Note

Depending on your implementation team, individual users may be combined and their responsibilities and authorizations may be assigned to a single user within the system.
5.2 Transportable Objects

In general, all SAP HANA Smart Data Integration and SAP HANA Smart Data Quality related objects are classified as SAP Core Data Services (CDS) artifacts and SQL artifacts. Transportability depends on the type of each object.

CDS Artifacts

<table>
<thead>
<tr>
<th>Object</th>
<th>Transportable</th>
</tr>
</thead>
<tbody>
<tr>
<td>Virtual table in flowgraph</td>
<td>Yes</td>
</tr>
<tr>
<td>Virtual table in replication task</td>
<td>Not applicable. Created during task activation; manual creation is not required.</td>
</tr>
<tr>
<td>Sequence</td>
<td>Yes</td>
</tr>
<tr>
<td>Target table in flowgraph</td>
<td>Yes</td>
</tr>
<tr>
<td>Procedure</td>
<td>Yes</td>
</tr>
<tr>
<td>Function</td>
<td>Yes</td>
</tr>
<tr>
<td>Table type</td>
<td>Yes</td>
</tr>
<tr>
<td>Calculation view</td>
<td>Yes</td>
</tr>
<tr>
<td>Database view</td>
<td>Yes</td>
</tr>
<tr>
<td>Flowgraph</td>
<td>Yes</td>
</tr>
<tr>
<td>Replication task</td>
<td>Yes</td>
</tr>
</tbody>
</table>

SQL Artifacts

<table>
<thead>
<tr>
<th>Object</th>
<th>Transportable</th>
</tr>
</thead>
<tbody>
<tr>
<td>Remote source</td>
<td>No</td>
</tr>
<tr>
<td>Virtual table in flowgraph</td>
<td>No</td>
</tr>
<tr>
<td>Virtual table in replication task</td>
<td>Not applicable. Created during task activation; manual creation is not required.</td>
</tr>
<tr>
<td>Sequence</td>
<td>No</td>
</tr>
<tr>
<td>Target table in flowgraph</td>
<td>No</td>
</tr>
</tbody>
</table>

See SAP Note 2414711 for more information.

5.3 Scenario

SDI_ADMIN, SDI_DEV, SDI_TRANSPORT, and SDI_EXEC have agreed to collaborate to create, review, transport, and execute replication tasks and flowgraphs in a common data mart project.
Limitations

If there are multiple developers in your project team, they require the same authorizations. However, only one user can work on a specific flowgraph or replication task at any given time.

Procedure

1. The SAP HANA administrator creates the users and assigns the following authorizations and roles:

<table>
<thead>
<tr>
<th>User</th>
<th>Authorization Type</th>
<th>Authorization Details</th>
</tr>
</thead>
</table>
| SDI_ADMIN     | System privileges    | CREATE REMOTE SOURCE
                |                      | CREATE SCHEMA
| Roles         |                      | sap.hana.xs.ide.roles::CatalogDeveloper
                |                      | sap.hana.xs.ide.roles::EditorDeveloper
| Package privileges |                    | REPO.READ on "public"
                |                      | REPO.MAINTAIN_IMPORTED_PACKAGES on "public"
                |                      | REPO.EDIT_NATIVE_OBJECTS on "public"
                |                      | REPO.ACTIVATE_NATIVE_OBJECTS on "public"
                |                      | REPO.MAINTAIN_NATIVE_PACKAGES on "public"
| Note          |                      | The SDI_ADMIN package privileges should be assigned with the GRANT OPTION.     |

| SDI_DEV       | Roles                | sap.hana.xs.ide.roles::CatalogDeveloper
                |                      | sap.hana.xs.ide.roles::EditorDeveloper

| SDI_TRANSPORT | Roles                | sap.hana.xs.lm.roles::Administrator
| SDI_EXEC      | Roles                | sap.hana.xs.ide.roles::CatalogDeveloper
                |                      | sap.hana.xs.ide.roles::EditorDeveloper

The SAP HANA administrator can use CALL GRANT_ACTIVATED_ROLE to assign the roles to each user:

```
CALL GRANT_ACTIVATED_ROLE('<role>', '<user>')</n```

2. SDI_ADMIN creates the remote source, package, and schemas, and assigns authorizations for the source and target schemas.

1. Create the RS_ECC remote source to be used by SDI_DEV.
   1. Log into the SAP HANA Web-based Development Workbench.
   2. Open the catalog and create remote source RS_ECC.

```
http://<hana_server>:80<instance>/sap/hana/ide/catalog
```

3. Provide authorization to use the remote source to SDI_DEV.
GRANT CREATE VIRTUAL TABLE, CREATE REMOTE SUBSCRIPTION, PROCESS REMOTE SUBSCRIPTION EXCEPTION, ALTER ON REMOTE SOURCE "RS_ECC" to SDI_DEV;

2. Create the PKG_DATAMART package for the data mart subject area.
   
   1. Open the editor and create package PKG_DATAMART under "public".  
      http://<hana_server>:80<instance>/sap/hana/ide/editor
   
   2. Provide authorization to access edit, activate, and maintain objects in the package to SDI_DEV.  
      GRANT REPO.READ, REPO.EDIT_NATIVE_OBJECTS, REPO.ACTIVATE_NATIVE_OBJECTS,  
      REPO.MAINTAIN_NATIVE_OBJECTS ON "public.PKG_DATAMART" TO SDI_DEV;

3. Create source and target schemas and allow SDI_DEV to access the schemas.
   
   o CREATE SCHEMA "SOURCE";
   o CREATE SCHEMA "TARGET";
   o GRANT SELECT, CREATE ANY ON "SOURCE" TO SDI_DEV;
   o GRANT SELECT, CREATE ANY ON "TARGET" TO SDI_DEV;
   
   SDI_DEV requires authorization to create virtual tables in the source schema and SAP HANA target tables in the target schema.

4. Grant _SYS_REPO authorizations required for SDI_DEV to activate flowgraphs and replication tasks on the development system.
   
   o GRANT SELECT ON SCHEMA "SOURCE" TO _SYS_REPO WITH GRANT OPTION;
   o GRANT CREATE VIRTUAL TABLE, CREATE REMOTE SUBSCRIPTION ON REMOTE SOURCE "RS_ECC" TO _SYS_REPO;

   1. Note

   If SDI_DEV needs to also execute flowgraphs or replication tasks in addition to activating them, they require the same authorizations as SDI_EXEC.

5. Authorize SDI_EXEC to execute flowgraphs and replication tasks.
   
   o GRANT SELECT, EXECUTE, ALTER ON SCHEMA "SOURCE" TO SDI_EXEC;
   o GRANT SELECT, EXECUTE, ALTER, INSERT, UPDATE, DELETE 
      ON SCHEMA "TARGET" TO SDI_EXEC;
   o GRANT REPO.READ, REPO.EDIT_IMPORTED_OBJECTS, REPO.ACTIVATE_IMPORTED_OBJECTS,  
      REPO.MAINTAIN_IMPORTED_OBJECTS ON "public.PKG_DATAMART" TO SDI_EXEC;

3. SDI_DEV creates replication task and flowgraphs.
   
   1. Log into the SAP HANA Web-based Development Workbench.
   
   2. Open the editor and create the flowgraph or replication task using the remote source, package, and source and target schemas provided by SDI_ADMIN.
4. SDI_TRANSPORT exports the package as a delivery unit and imports the delivery unit into the destination system.

1. In SAP HANA Application Lifecycle Management, create a delivery unit containing the transportable objects from the source system.

2. In the destination system, create non-transportable objects with the same characteristics as the original objects in the source system. For example:
   - Remote sources with the same names
   - Virtual tables and target tables with the same schema and table names for flowgraphs

3. In SAP HANA Application Lifecycle Management, import the delivery unit into the destination system.
   
   If dependent objects have not already been created in the destination system, the import will fail. In this specific case, you can import the delivery unit into the destination system by using SAP HANA studio and unchecking the **Activate objects** option.

   **Note**

   Because importing the delivery unit also activates flowgraphs and replication tasks, the following additional authorizations must be granted in the destination system:

<table>
<thead>
<tr>
<th>User</th>
<th>Authorization Details</th>
</tr>
</thead>
</table>
   | SDI_TRANSPORT | • SELECT ON SCHEMA "SOURCE"
                 | • EXECUTE ON SCHEMA "SOURCE"                                                           |
   | _SYS_REPO     | • SELECT, INSERT, UPDATE, DELETE ON SCHEMA "SOURCE" WITH GRANT OPTION                  |
   |               | • CREATE VIRTUAL TABLE, CREATE REMOTE SUBSCRIPTION ON REMOTE SOURCE "RS_ECC"           |

5. SDI_EXEC executes the flowgraphs and replication tasks in the destination system.

1. Log into the SAP HANA Web-based Development Workbench.

2. Open the editor and execute the flowgraph or replication task.

   ```
   http://<hana_server>:80<instance>/sap/hana/ide/editor
   ```
6 Tuning Guidelines

This section provides generic guidelines for tuning various configurations in an SAP HANA smart data integration replication landscape. The tuning information provided is based on HANA 1.0 SPS12 Rev 122.05+ with Data Provisioning Agent 1.3.4+. You can start with these recommendations, run your work load in your own landscape and perform additional tuning based on your test results.

Although certain parameters need to be tuned on the SAP HANA server, most sizing and tuning is done on the Data Provisioning Agent. For the Data Provisioning Agent, sizing refers to memory (RAM) in GB, number of CPU cores, storage filesystem, and network bandwidth requirements for inbound and outbound communications.

Tuning Scenarios

The tuning parameters that are applicable depend on the workload or scenario and how replication and certain data transformations are performed.

6.1 Initial Load/Batch Replication

In initial load or batch replication, a large amount of data is replicated once or on a periodic basis. During loading, the Data Provisioning Agent reads and forwards the data to Data Provisioning Server. The primary configurable parameters that can impact performance during this process are the Java heap size and fetch size for the Data Provisioning Agent process.

6.1.1 Sample Data Provisioning Agent Tuning

For large initial loads, in dpagent.ini, configure for 16GB or 24GB of memory

- Xmx16384m
- Xmx24576m

6.1.2 Review Performance and Identify Bottlenecks

To start tuning the performance of a replication task, perform an execution of your replication task and identify any bottlenecks.

1. Design and execute your replication task.
2. In the Data Provisioning Task Monitor, select the task in the Task Overview table.

3. Select the initial execution of the task from the Task Execution Monitor table.

4. Identify bottlenecks from the information in the Task Operation Execution Monitor table.
   Look for operations with a significantly longer duration than other options. In most cases, reading the source data is often the initial bottleneck.

   ![Note]

   If the replication task is taking a long time to read the source data, the task operation execution statistics may not appear until the reading operation has finished. To monitor the progress of the reader, use the SAP HANA Smart Data Access monitor. For more information, see "Monitor Remote Connections and Remote Statements" in the SAP HANA Administration Guide.

For additional information about the types of statistics available in the Data Provisioning Task Monitor, see "Monitoring Data Provisioning Tasks" in the Administration Guide for SAP HANA Smart Data Integration and SAP HANA Smart Data Quality.

6.1.3 Task Partitioning

To improve performance for large source objects, consider using the task partitioning option. Task partitioning allows SAP HANA to read, process and commit the partitioned virtual table input sources in parallel.

Define task partitions in the Partitions tab within the replication editor. Two partition types are available: range partitions and list partitions.

- Range partitions may specify only a single value.
  Range partitioning is easier if there is a key value that can be used, especially something sequential.

For example, if you have a column named payment_detail_id that contains sequential IDs, select that column as the range attribute and create partitions based on ID increments:

<table>
<thead>
<tr>
<th>Partition Name</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Part 1</td>
<td>500000000</td>
</tr>
<tr>
<td>Part 2</td>
<td>100000000</td>
</tr>
<tr>
<td>Part 3</td>
<td>150000000</td>
</tr>
<tr>
<td>Part 4</td>
<td>200000000</td>
</tr>
</tbody>
</table>

- List partitions may specify either a single value or a comma delimited list of values.
List partitioning requires knowledge of the values being used in the specified column, and may require introspection of the source data.

For example, if you have source data including a `month` column, select that column as the list attribute and create partitions based on the months that you want to include:

<table>
<thead>
<tr>
<th>Partition Name</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Q1</td>
<td>'1', '2', '3'</td>
</tr>
<tr>
<td>Q2</td>
<td>'4', '5', '6'</td>
</tr>
<tr>
<td>Q3</td>
<td>'7', '8', '9'</td>
</tr>
<tr>
<td>Q4</td>
<td>'10', '11', '12'</td>
</tr>
</tbody>
</table>

- Task partitions may be executed sequentially or concurrently, depending on the priority of execution time versus memory consumption.

The value entered for ‘Number of parallel partitions’ determines the number of partitions that will execute in parallel.

A value of 1 implies that all partitions will execute sequentially. This will be slower from an execution time perspective, but will limit the amount of memory consumed on the target HANA system because data is committed and flushed from temporary memory after each partition.

A value >1 implies that this is the number of parallel partitions to be executed. It cannot be greater than the total number of partitions. If it is equal to the total number of partitions, all partitions will be executed in parallel, giving the best execution time for the scenario, but also requiring the most available memory.

**Note**

When you use task partitioning, the statistics that appear in the Task Overview and Task Execution Monitor tables of the Data Provisioning Task Monitor are unchanged. However, in the Task Operations Execution Monitor table, an individual operation node for each task partition appears.

### 6.1.4 Target Table Partitioning

When the Table Writer becomes the bottleneck, consider partitioning the target table. Target table partitioning is unsupported in the replication editor, and must be done after the replication task has been saved and activated (and the table is created/truncated), but before the replication task is executed.

Round robin and hash are the most relevant partitioning methods for target table partitioning. Round robin partitioning generally produces the best performance and memory profile, but can only be used when there are no primary key values on the table. Hash partitions may be used when the target table contains a primary key.

1. Create and save a replication task with the selected remote source objects.
2. For each remote source object, browse to the SAP HANA target schema and alter the table to add hash or round robin partitioning. For example:

```sql
ALTER TABLE "<target_table>" PARTITION BY ROUNDROBIN PARTITIONS <number_of_partitions>;
ALTER TABLE "<target_table>" PARTITION BY HASH (<primary_key_columns>) PARTITIONS <number_of_partitions>;
```

3. Execute the replication task from the editor.

For additional information about table partitioning, see the SAP HANA Administration Guide. For more detailed information about the SQL syntax for partitioning, see the SAP HANA SQL and System Views Reference.

### 6.2 Real-time Replication

In real-time replication, changes are captured and distributed to the target system. When the changes are captured from transaction logs, tuning is performed to scan the logs in parallel and scanning parameters are adjusted to reduce latency in replication.

There are three key performance indicators of a replication agent:

- **Latency** - The amount of time between when an operation is executed against the primary database and when the operation is replicated by the Data Provisioning Agent.
- **Throughput** - The amount of data that flows through the Data Provisioning Agent in a given time period.
- **Resource Usage** - How many resources (CPU, RAM, etc.) the Data Provisioning Agent consumes during replication.

#### 6.2.1 Measuring Latency

The best way to measure agent latency is to use the LATENCY MONITORING capability. When LATENCY MONITORING is turned on, a ticket executed at the primary database will be written to the instance log after being processed by the LogReader Adapter. The ticket will also be stored in M_REMOTE_SOURCE_LATENCY_HISTORY view in the target HANA database.

LATENCY MONITORING entries in M_REMOTE_SOURCE_LATENCY_HISTORY view will contain the timestamp when the trace was written to the instance log and the time the ticket was executed at the primary database. The difference between the two timestamps tells you the Data Provisioning Agent latency.

When you measure latency, observe trends and not momentary snapshots. For example, is the agent falling behind or catching up? If it is catching up, performance may be sufficient and no additional tuning may be needed. If it is falling behind and latency is increasing, you may consider tuning your replication agent.
You can measure trends by issuing LATENCY MONITORING at regular intervals and then searching M_REMOTE_SOURCE_LATENCY_HISTORY view for LATENCY MONITORING entries.

### 6.2.2 Throughput

Requirements on Data Provisioning Agent throughput are dictated by the log generation rate of the primary database. Determining the log generation rate differs for each source database.

For example, on Oracle, the log generation rate can be determined by querying the `v$archived_log` view.

### 6.2.3 Resource Usage

The Data Provisioning Agent consumes resources directly while processing log records, using memory and CPU cycles to read, parse, evaluate, format and send data.

The resource usage profile differs based on the log reading technology:

- The Oracle LogReader Adapter consumes Oracle resources indirectly by accessing Oracle’s LogMiner interface.
- The DB2 LogReader Adapter consumes UDB resources indirectly through the `db2readlog` API.
- The MSSQL LogReader Adapter consumes negligible database server resources because it reads the MS SQL Server log file directly. The only database server resources it consumes are for operations such as log truncation.

Direct memory usage can be measured using the Data Provisioning Agent Remote Source Statistics. If `VM % max memory used` consistently exceeds 50%, consider increasing the amount of memory available to the agent.

Direct CPU Usage of the Data Provisioning Agent JVM can be measured using a number of operating system utilities such as `vmstat` on Linux platforms and Process Explorer on Windows.

**Oracle LogReader Adapter-specific Resource Usage**

You can measure indirect resource usage of the Data Provisioning Agent by using Oracle tools such as the Automatic Workload Repository (AWR).

Complete details about AWR reports are outside the scope of this document, but common areas where replication can place load on the Oracle database include:
- Database CPU time
- Log file sequential read time
- SQL query execution time
- System idle time

To determine the resource load generated by replication, run Oracle with the replication agent suspended for a time interval while the Oracle system is under a representative workload. Then, do the same with a tuned replication agent running.

Use AWR reports to compare Oracle resource usage for the times when the Data Provisioning Agent is suspended and when it is running. The difference in the two usage profiles highlights the load that LogMiner used by the Data Provisioning Agent places on the Oracle server.

### 6.2.4 Tuning Customization

Generally, the default configuration values provide optimal performance. However, there may be certain situations where the configuration should be changed to suit or optimize your particular environment.

#### 6.2.4.1 Adjusting the Replication Agent System Logs

By default, system logs produced by the LogReader Adapters are a fixed size and roll over occasionally to prevent limitless disk consumption.

To save log data for a longer period of time, increase *Maximum log file size* and adjust *Number of wrapped log files* in the LogReader Adapter preferences. Decreasing these parameters will reduce the amount of space consumed in your environment.

#### 6.2.4.2 Tuning Parallel Scanner Configuration

Parallel scanning is supported by only by the Oracle LogReader Adapter. Parallel scanning can increase replication throughput by executing multiple queries against the primary database LogMiner on separate threads and serializing the results. Optimizing a parallel scanner configuration requires careful tuning.

First, in the Oracle LogReader Adapter, check that you have enough parallel scanners (*Number of parallel scanners*), the right scan range (*SCN range of parallel scan tasks*) for each and sufficiently large parallel scan queues (*Queue size of parallel scan tasks*).

With an insufficient number of parallel scanners, the adapter must wait for query results from LogMiner. With a scan range that is too large, the adapter may idle LogMiner scanners while it drains their queues.
Many factors contribute to determining whether a throughput requirement is small, medium or large, including transaction profile and resource availability. For example, 10 GB/HR of archive log generation on one system with one schema may be far more demanding than a similar throughput requirement on a different host that has more and faster processors and a different schema.

However, for the sake of simplicity, consider the following as throughput requirements of systems and their suggested starting tuning parameters:

<table>
<thead>
<tr>
<th>System Size</th>
<th>Throughput</th>
<th>Parallel Scanners</th>
<th>SCN Range</th>
<th>Scan Queue Size</th>
</tr>
</thead>
<tbody>
<tr>
<td>Small</td>
<td>10 GB/HR</td>
<td>2</td>
<td>4000</td>
<td>100,000</td>
</tr>
<tr>
<td>Medium</td>
<td>30 GB/HR</td>
<td>5</td>
<td>2000</td>
<td>100,000</td>
</tr>
<tr>
<td>Large</td>
<td>50 GB/HR</td>
<td>10</td>
<td>1000</td>
<td>100,000</td>
</tr>
</tbody>
</table>

The recommended scan queue size depends on the amount of memory available to the JVM:

<table>
<thead>
<tr>
<th>System Size</th>
<th>Available JVM Memory</th>
<th>Recommended Scan Queue Size</th>
</tr>
</thead>
<tbody>
<tr>
<td>Small</td>
<td>4 GB</td>
<td>10,000</td>
</tr>
<tr>
<td>Medium</td>
<td>15 GB</td>
<td>50,000</td>
</tr>
<tr>
<td>Large</td>
<td>30 GB</td>
<td>100,000</td>
</tr>
</tbody>
</table>

The scan queue size may need to be adjusted based on the number of scanners, the SCN range and available memory. For example, with 10 scanners, an SCN range of 1000 and 30GB of memory, a queue size of 100K may be appropriate, but for the same SCN range and memory, but 20 scanners, 100K might be too large.

One indicator that the scan queue size may be too large is VM % max memory used regularly at 50% or greater in the Data Provisioning Agent Remote Source Statistics. When this occurs, consider either reducing the scan queue size or increasing the amount of memory allocated to the Data Provisioning Agent. For more information about controlling the JVM memory configuration, see Java VM statistics.

To determine sufficient queue size, review the following statistics:

- **Scan Queue Wait time**
  Amount of time the parallel scanners have spent waiting to queue their results due to a full queue. During this time, the scanners cannot return to query LogMiner for more log records.

- **Scan Queue Poll time**
  Amount time the LogReader spends polling for results from the scanner queue because the queue is empty.
A wait time near 0 with poll time increasing indicates that the scanner queue size is sufficient. If wait time is increasing, then the SCN range may be too large (i.e. too many results to fit in the queue) or the queue size may be too small. Adjust the scan range and the queue size accordingly.

If latency is too high, you generally want to increase the number of parallel scanners. As you increase the number of scanners, consider decreasing the scan range. If latency is lower than you need it to be, consider reducing the number of scanners and increasing the scan range in order to reduce load on the primary system.

In addition to latency, compare the following statistics:

- **LogMiner Query Time**
  
  Amount of time spent selecting data from the LogMiner.

- **LogMiner Process Time**
  
  Amount of time spent writing data to the parallel scan queues.

If the process time is significantly larger than the query time, try decreasing the scan range. If the query time significantly exceeds the process time, consider increasing the scan range.

Increasing wait or poll times may also be a side effect of the LogReader encountering an unreadable log record when handling LogMiner results. When encountering a record it cannot read, it will return a log record indicating that it could not read the record. For example, LogMiner will return an “Unsupported Operation” value for a log record that is part of a chained operation if the log record crosses a parallel scan boundary.

When the LogReader encounters an “Unsupported Operation” message, it launches a standby scanner to rescan the log for the offending record to be sure no replicable data has been missed. It must then pause the queuing of processed log records until the standby scanner returns in order to preserve the correct operation order. Standby scanner activity can be identified by checking standby scan frequency in the statistics.

If there are numerous standby scan calls with significant standby scan time and you find “Unsupported Operation” exception warnings in the LogReader instance log, you may be able to decrease standby scan frequency by increasing SCN range. An increased SCN range reduces the likelihood that a parallel scanner query boundary will land in the middle of a chained operation.