

Oracle® Fusion Middleware

Application Adapters Guide for Oracle Data Integrator

12c (12.1.3)

E51088-02

January 2015

Copyright © 2010, 2015, Oracle and/or its affiliates. All rights reserved.

Primary Author: Laura Hofman Miquel, Aslam Khan

This software and related documentation are provided under a license agreement containing restrictions on use and disclosure and are protected by intellectual property laws. Except as expressly permitted in your license agreement or allowed by law, you may not use, copy, reproduce, translate, broadcast, modify, license, transmit, distribute, exhibit, perform, publish, or display any part, in any form, or by any means. Reverse engineering, disassembly, or decompilation of this software, unless required by law for interoperability, is prohibited.

The information contained herein is subject to change without notice and is not warranted to be error-free. If you find any errors, please report them to us in writing.

If this is software or related documentation that is delivered to the U.S. Government or anyone licensing it on behalf of the U.S. Government, then the following notice is applicable:

U.S. GOVERNMENT END USERS: Oracle programs, including any operating system, integrated software, any programs installed on the hardware, and/or documentation, delivered to U.S. Government end users are "commercial computer software" pursuant to the applicable Federal Acquisition Regulation and agency-specific supplemental regulations. As such, use, duplication, disclosure, modification, and adaptation of the programs, including any operating system, integrated software, any programs installed on the hardware, and/or documentation, shall be subject to license terms and license restrictions applicable to the programs. No other rights are granted to the U.S. Government.

This software or hardware is developed for general use in a variety of information management applications. It is not developed or intended for use in any inherently dangerous applications, including applications that may create a risk of personal injury. If you use this software or hardware in dangerous applications, then you shall be responsible to take all appropriate fail-safe, backup, redundancy, and other measures to ensure its safe use. Oracle Corporation and its affiliates disclaim any liability for any damages caused by use of this software or hardware in dangerous applications.

Oracle and Java are registered trademarks of Oracle and/or its affiliates. Other names may be trademarks of their respective owners.

Intel and Intel Xeon are trademarks or registered trademarks of Intel Corporation. All SPARC trademarks are used under license and are trademarks or registered trademarks of SPARC International, Inc. AMD, Opteron, the AMD logo, and the AMD Opteron logo are trademarks or registered trademarks of Advanced Micro Devices. UNIX is a registered trademark of The Open Group.

This software or hardware and documentation may provide access to or information about content, products, and services from third parties. Oracle Corporation and its affiliates are not responsible for and expressly disclaim all warranties of any kind with respect to third-party content, products, and services unless otherwise set forth in an applicable agreement between you and Oracle. Oracle Corporation and its affiliates will not be responsible for any loss, costs, or damages incurred due to your access to or use of third-party content, products, or services, except as set forth in an applicable agreement between you and Oracle.

Contents

Preface	ix
Audience.....	ix
Documentation Accessibility	ix
Related Documents	ix
Conventions	x
1 Introduction	
1.1 Terminology.....	1-1
1.2 Using This Guide	1-2
2 Oracle E-Business Suite	
2.1 Introduction	2-1
2.1.1 Concepts.....	2-1
2.1.2 Knowledge Modules	2-2
2.2 Installation and Configuration.....	2-2
2.2.1 System Requirements and Certifications	2-2
2.2.2 Technology Specific Requirements	2-3
2.2.3 Connectivity Requirements.....	2-3
2.3 Setting Up the Topology	2-3
2.3.1 Create an Oracle Data Server	2-3
2.3.2 Create an Oracle Physical Schema	2-3
2.4 Setting up an Integration Project	2-4
2.5 Creating an Oracle Model and Reverse-Engineering E-Business Suite Tables.....	2-4
2.5.1 Create an Oracle Model	2-4
2.5.2 Reverse-Engineer E-Business Suite Tables.....	2-4
2.6 Designing a Mapping	2-6
2.6.1 Loading Data from E-Business Suite	2-6
2.6.2 Integrating Data in E-Business Suite through the Open Interface	2-6
2.6.2.1 Managing Group IDs	2-7
2.6.2.2 Executing an Open Interface Program	2-8
3 Hadoop	
3.1 Introduction	3-1
3.1.1 Concepts.....	3-2
3.1.2 Knowledge Modules	3-2

3.2	Installation and Configuration.....	3-3
3.2.1	System Requirements and Certifications	3-3
3.2.2	Technology-Specific Requirements.....	3-3
3.2.3	Location of Oracle Data Integrator Application Adapter for Hadoop	3-4
3.3	Setting Up the Topology	3-4
3.3.1	Setting Up File Data Sources.....	3-4
3.3.2	Setting Up Hive Data Sources.....	3-5
3.3.3	Setting Up HBase Data Sources.....	3-5
3.3.4	Connecting to a Secure Cluster.....	3-6
3.3.5	Setting Up the Oracle Data Integrator Agent to Execute Hadoop Jobs.....	3-7
3.3.6	Configuring Oracle Data Integrator Studio for Executing Hadoop Jobs on the Local Agent 3-9	
3.4	Setting Up an Integration Project	3-9
3.5	Creating an Oracle Data Integrator Model from a Reverse-Engineered Hive and HBase Model 3-10	
3.5.1	Creating a Model	3-10
3.5.2	Reverse Engineering Hive Tables.....	3-10
3.5.3	Reverse Engineering HBase Tables.....	3-11
3.6	Designing a Mapping	3-12
3.6.1	Loading Data from Files into Hive.....	3-12
3.6.2	Loading Data from HBase into Hive	3-15
3.6.3	Loading Data from Hive into HBase	3-16
3.6.4	Loading Data from an SQL Database into Hive, HBase, and File.....	3-17
3.6.5	Validating and Transforming Data Within Hive	3-19
3.6.5.1	IKM Hive Control Append	3-19
3.6.5.2	CKM Hive.....	3-19
3.6.5.3	IKM Hive Transform.....	3-20
3.6.6	Loading Data into an Oracle Database from Hive and HDFS	3-22
3.6.7	Loading Data into a SQL Database from Hive and HDFS	3-25

4 JD Edwards World

4.1	Introduction	4-1
4.1.1	Concepts.....	4-1
4.1.2	Knowledge Modules	4-1
4.2	Installation and Configuration.....	4-2
4.2.1	System Requirements and Certifications	4-2
4.2.2	Technology Specific Requirements	4-2
4.2.3	Connectivity Requirements.....	4-3
4.3	Setting Up the Topology	4-4
4.3.1	Create a Data Server	4-4
4.3.2	Create a Physical Schema	4-4
4.4	Setting up an Integration Project	4-5
4.5	Creating and Reverse-Engineering a Model.....	4-5
4.5.1	Create a Model	4-5
4.5.2	Reverse-Engineer JDE Tables.....	4-5
4.6	Designing a Mapping	4-6
4.6.1	Loading Data from JDE.....	4-7

4.6.2	Integrating Data in JDE.....	4-7
-------	------------------------------	-----

5 Oracle PeopleSoft

5.1	Introduction	5-1
5.1.1	Concepts.....	5-1
5.1.2	Knowledge Modules	5-1
5.2	Installation and Configuration.....	5-2
5.2.1	System Requirements and Certifications	5-2
5.2.2	Technology Specific Requirements	5-2
5.2.3	Connectivity Requirements.....	5-2
5.3	Setting up the Topology.....	5-2
5.3.1	Create a Data Server	5-2
5.3.2	Create a Physical Schema	5-3
5.4	Setting up the Project.....	5-3
5.5	Creating and Reverse-Engineering a Model.....	5-3
5.5.1	Create a Model	5-3
5.5.2	Reverse-Engineer PeopleSoft Tables.....	5-3
5.6	Designing a Mapping	5-4
5.6.1	Loading Data from PeopleSoft.....	5-4

6 SAP ABAP BW

6.1	Introduction	6-1
6.1.1	Concepts.....	6-1
6.1.2	Knowledge Modules	6-2
6.1.3	Overview of the SAP BW Integration Process.....	6-2
6.1.3.1	Reverse-Engineering Process.....	6-3
6.1.3.2	Integration Process	6-3
6.2	Installation and Configuration.....	6-3
6.2.1	System Requirements and Certifications	6-3
6.2.2	Technology Specific Requirements	6-4
6.2.3	Connectivity Requirements.....	6-5
6.2.3.1	Installing and Configuring JCo	6-5
6.2.3.2	Installing ODI SAP Components into SAP System	6-6
6.2.3.3	Validating the SAP Environment Setup.....	6-6
6.2.3.4	Gathering SAP Connection Information.....	6-6
6.2.3.5	Gathering FTP Connection Information	6-8
6.2.3.6	Gathering Shared Directory Information.....	6-8
6.2.4	Adding the Open Tool	6-8
6.3	Defining the Topology	6-9
6.3.1	Create the File Data Server	6-9
6.3.1.1	Create a File Data Server	6-9
6.3.1.2	Create the File Schema.....	6-10
6.3.2	Create the SAP ABAP Data Server.....	6-11
6.3.2.1	Create the SAP ABAP Data Server	6-11
6.3.2.2	Create the SAP ABAP Schema.....	6-13
6.4	Setting up the Project.....	6-13

6.5	Creating and Reverse-Engineering a Model.....	6-13
6.5.1	Creating a SAP BW Model	6-13
6.5.2	Reverse-Engineering a SAP BW Model.....	6-13
6.6	Designing a Mapping	6-14
6.7	Considerations for SAP BW Integration.....	6-14
6.7.1	File Transfer Configurations	6-14
6.7.1.1	Transfer using a Shared Directory (recommended).....	6-14
6.7.1.2	FTP based Transfer.....	6-16
6.7.2	Execution Modes.....	6-18
6.7.3	Controlling ABAP Uploading / ABAP code in production	6-18
6.7.4	Managing ODI SAP Transport Requests.....	6-20
6.7.5	SAP Packages and SAP Function Groups	6-21
6.7.6	Log Files	6-22
6.7.7	Limitation of the SAP BW Adapter.....	6-22

7 SAP ABAP ERP

7.1	Introduction	7-1
7.1.1	Concepts.....	7-1
7.1.2	Knowledge Modules	7-1
7.1.3	Overview of the SAP ABAP Integration Process	7-2
7.1.3.1	Reverse-Engineering Process.....	7-2
7.1.3.2	Integration Process	7-2
7.2	Installation and Configuration.....	7-3
7.2.1	System Requirements and Certification.....	7-3
7.2.2	Technology Specific Requirements	7-3
7.2.3	Connectivity Requirements.....	7-4
7.2.3.1	Installing and Configuring JCo	7-4
7.2.3.2	Installing ODI SAP Components into SAP System	7-5
7.2.3.3	Validating the SAP Environment Setup.....	7-5
7.2.3.4	Gathering SAP Connection Information	7-6
7.2.3.5	Gathering FTP Connection Information	7-7
7.2.3.6	Gathering Shared Directory Information.....	7-7
7.2.4	Adding the Open Tool	7-8
7.3	Defining the Topology	7-8
7.3.1	Create the File Data Server	7-8
7.3.1.1	Create a File Data Server	7-8
7.3.1.2	Create the File Schema.....	7-9
7.3.2	Create the SAP ABAP Data Server.....	7-10
7.3.2.1	Create the SAP ABAP Data Server.....	7-10
7.3.2.2	Create the SAP ABAP Schema.....	7-12
7.4	Setting up the Project.....	7-12
7.5	Creating and Reverse-Engineering a Model.....	7-12
7.5.1	Creating a SAP ERP Model	7-13
7.5.2	Reverse-Engineering a SAP ERP Model.....	7-13
7.6	Designing a Mapping	7-13
7.7	Considerations for SAP ERP Integration.....	7-13
7.7.1	File Transfer Configurations	7-14

7.7.1.1	Transfer using a Shared Directory (recommended).....	7-14
7.7.1.2	FTP based Transfer.....	7-16
7.7.2	Execution Modes.....	7-17
7.7.3	Controlling ABAP Uploading	7-18
7.7.4	Managing ODI SAP Transport Requests.....	7-19
7.7.5	SAP Packages and SAP Function Groups	7-20
7.7.6	Log Files	7-21
7.7.7	Limitations of the SAP ABAP Adapter	7-22

A Additional Information for SAP ABAP BW Adapter

A.1	SAP ABAP BW Required Privileges	A-1
A.1.1	Important points to consider.....	A-2
A.1.2	Authorizations Required for RKM SAP BW Setup	A-2
A.1.3	Authorizations Required for RKM SAP BW Execution	A-3
A.1.4	Authorizations Required for LKM SAP BW Execution	A-4
A.1.5	Authorizations Required for LKM SAP BW Execution in Production	A-5
A.1.6	Authorizations Required for LKM SAP BW Execution as Background Process.....	A-6
A.1.7	Authorizations Required for LKM SAP BW Execution as Background Process in Production A-7	
A.2	SAP Stand-Alone Connection Test.....	A-8
A.2.1	SAP Stand-Alone Connection Test.....	A-8

B Additional Information for SAP ABAP ERP Adapter

B.1	SAP ABAP ERP Required Privileges	B-1
B.1.1	Important points to consider.....	B-2
B.1.2	Authorizations Required for RKM SAP ERP Setup	B-2
B.1.3	Authorizations Required for RKM SAP ERP Execution	B-3
B.1.4	Authorizations Required for LKM SAP ERP Execution	B-3
B.1.5	Authorizations Required for LKM SAP ERP Execution in Production.....	B-4
B.1.6	Authorizations Required for LKM SAP ERP Execution as Background Process.....	B-5
B.1.7	Authorizations Required for LKM SAP ERP Execution as Background Process in Production B-6	
B.2	SAP Connection Test	B-6
B.3	SAP Stand-Alone Connection Test.....	B-7

C Installing ODI SAP Components

C.1	Updating ODI SAP Components.....	C-1
C.2	Installing ODI SAP Components.....	C-1
C.2.1	Installing SAP Transport Request (TR)	C-2
C.2.1.1	Downloading the Transport Request files	C-2
C.2.1.2	Installing the Transport Request Files	C-2
C.2.2	Installing and Assigning SAP User Profile	C-6
C.3	Validating the ODI SAP Setup	C-8
C.3.1	Validating the Shared Folder Setup	C-8
C.3.2	Validating the FTP Setup.....	C-9
C.3.3	Validating SAP Privileges	C-10

C.3.4	Validating SAP Transport Layer Name	C-10
C.4	Uninstalling ODI SAP Components.....	C-11
C.4.1	Validating Uninstallation of ODI SAP Components.....	C-12

D Moving ODI and SAP Components from Development to Production

D.1	Transport Request (TR).....	D-1
D.1.1	Viewing the List of TRs Created after RKM Execution.....	D-2
D.1.1.1	Description of TR Fields	D-3
D.1.2	Transport Request (TR) relevant to DEV	D-3
D.1.3	Transport Request (TR) relevant to PROD.....	D-3
D.2	Remote Function Call (RFC).....	D-4
D.3	Adding Mappings Under Same Transport Request	D-5
D.3.1	KM Options Descriptions	D-7
D.4	Creating ODI Package.....	D-8
D.5	Generating ODI Scenario	D-9
D.6	Releasing Transport Request to Production	D-9
D.7	Releasing Transport Request of the Mapping to Production	D-11
D.8	Executing the ODI Scenario.....	D-13

Preface

This manual describes how to configure and work with the Application Adapters in Oracle Data Integrator.

This preface contains the following topics:

- [Audience](#)
- [Documentation Accessibility](#)
- [Related Documents](#)
- [Conventions](#)

Audience

This document is intended for developers and administrators who want to use Oracle Data Integrator Application Adapters as a development tool for their integration processes.

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.

Related Documents

For more information, see the following documents in *Oracle Data Integrator Library*.

- Release Notes for Oracle Data Integrator
- Understanding Oracle Data Integrator
- Administering Oracle Data Integrator
- Developing Integration Projects with Oracle Data Integrator
- Installing and Configuring Oracle Data Integrator
- Upgrading Oracle Data Integrator

- Developing Knowledge Modules with Oracle Data Integrator
- Connectivity and Knowledge Modules Guide for Oracle Data Integrator
- Migrating From Oracle Warehouse Builder to Oracle Data Integrator
- Oracle Data Integrator Tool Reference
- Data Services Java API Reference for Oracle Data Integrator
- Open Tools Java API Reference for Oracle Data Integrator
- Getting Started with SAP ABAP BW Adapter for Oracle Data Integrator
- Java API Reference for Oracle Data Integrator
- Getting Started with SAP ABAP ERP Adapter for Oracle Data Integrator
- *Oracle Data Integrator 12c Online Help*, which is available in ODI Studio through the JDeveloper Help Center when you press **F1** or from the main menu by selecting **Help**, and then **Search** or **Table of Contents**.

Conventions

The following text conventions are used in this document:

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

Introduction

This chapter introduces the terminology used in this guide and provides information about how to use the guide.

Oracle Data Integrator uses Application Adapters to integrate data from and to enterprise applications.

This book describes how to configure and work with the Application Adapters in Oracle Data Integrator for the following technologies:

- [Oracle E-Business Suite](#)
- [Hadoop](#)
- [JD Edwards World](#)
- [Oracle PeopleSoft](#)
- [SAP ABAP BW](#)
- [SAP ABAP ERP](#)

This chapter includes the following sections:

- [Section 1.1, "Terminology"](#)
- [Section 1.2, "Using This Guide"](#)

1.1 Terminology

This section defines some common terms that are used in this document and throughout the related documents mentioned in the [Preface](#).

Knowledge Module

Knowledge Modules (KMs) are components of Oracle Data Integrator' Open Connector technology. KMs contain the knowledge required by Oracle Data Integrator to perform a specific set of tasks against a specific technology or set of technologies.

Combined with a connectivity layer such as, for example, JDBC, JMS, or JCA, Knowledge Modules define an Open Connector that performs defined tasks against a technology, such as connecting to this technology, extracting data from it, transforming the data, checking it, integrating it, etc.

Application Adapter

An *adapter* is a group of Knowledge Modules. In some cases, this group also contains an attached technology definition for Oracle Data Integrator.

Oracle Application Adapters for Data Integration provide specific software components for reverse-engineering metadata from, and extracting bulk data from various applications.

1.2 Using This Guide

This guide provides conceptual information and processes for working with application adapters and technologies supported in Oracle Data Integrator.

Each chapter explains how to configure a given technology, set up a project and use the technology-specific application adapters to perform integration operations.

Some knowledge modules are not technology-specific and require a technology that support an industry standard. These knowledge modules are referred to as *Generic* knowledge modules and are designed to work respectively with any ANSI SQL-92 compliant database and any JMS compliant message provider. See "Generic SQL" in the *Connectivity and Knowledge Modules Guide for Oracle Data Integrator* for more information.

When these generic knowledge module can be used with a technology, the technology chapter will mention it. However, we recommend using technology-specific knowledge modules for better performances and enhanced technology-specific feature coverage.

Before using a knowledge module, it is recommended to review the knowledge module description in Oracle Data Integrator Studio for usage details, limitations and requirements. In addition, although knowledge modules options are pre-configured with default values to work out of the box, it is also recommended to review these options and their description.

The chapters in this guide will provide you with the important usage, options, limitation and requirement information attached to the technologies and application adapters.

Oracle E-Business Suite

This chapter describes how to work with Oracle E-Business Suite Knowledge Modules in Oracle Data Integrator.

This chapter includes the following sections:

- [Section 2.1, "Introduction"](#)
- [Section 2.2, "Installation and Configuration"](#)
- [Section 2.3, "Setting Up the Topology"](#)
- [Section 2.4, "Setting up an Integration Project"](#)
- [Section 2.5, "Creating an Oracle Model and Reverse-Engineering E-Business Suite Tables"](#)
- [Section 2.6, "Designing a Mapping"](#)

2.1 Introduction

Oracle E-Business Suite (EBS) is a suite of integrated software applications that provides a complete solution to the business needs of Oracle customers.

2.1.1 Concepts

The EBS Knowledge Modules provide support for the following capabilities:

- Reverse-engineering EBS objects: RKM E-Business Suite can be used to reverse-engineer E-Business Suite data structures.
- Data extraction from EBS: Standard Oracle or SQL LKMs can be used to extract data from E-Business suite using objects such as Tables, Views, and KeyFlexfields.
- Data integration to EBS: IKM E-Business Suite can be used to integrate data to E-Business Suite using Open Interface tables. The "Open Interface" API encapsulates a number of Oracle-specific interfaces and ensures data integrity. An Open Interface is made up of:
 - Several Interface tables to be loaded. These tables are the incoming data entry points for E-Business Suite.
 - Several programs that validate and process the insertion of the data from the interface tables into E-Business Suite.

Oracle Data Integrator Knowledge Modules for Oracle E-Business Suite interact with the database tier to extract metadata and load data. While loading data, it also interacts with the Concurrent Processing Server of the application tier.

2.1.2 Knowledge Modules

Oracle Data Integrator provides the Knowledge Modules listed in [Table 2–1](#) for handling E-Business Suite data. These specific EBS KMs provide comprehensive, bidirectional connectivity between Oracle Data Integrator and E-Business Suite, which enables you to extract and load data. The Knowledge Modules support all modules of E-Business Suite and provide bidirectional connectivity through EBS objects tables/views and interface tables.

Table 2–1 Oracle E-Business Suite Knowledge Modules

Knowledge Module	Description
IKM E-Business Suite (Open Interface)	<p>The IKM E-Business Suite is used to load data to EBS interface tables and submit Concurrent request (which loads from interface tables to base tables).</p> <p>This Integration Knowledge Module:</p> <ul style="list-style-type: none"> Integrates data from any source to Interface Tables in incremental update mode. Enables data control: invalid data is isolated in the Error Table and can be recycled. <p>In addition to loading the interface tables, it provides the following optional actions:</p> <ul style="list-style-type: none"> Create a Group ID for the first mapping in a batch. Use this Group ID in subsequent mappings. Delete this Group ID when loading the last table in the batch. Execute an Open Interface program if at any point in a batch it is required to call an E-Business Suite Interface program and once all required interface tables have been loaded. <p>Note that the IKM E-Business Suite (Open Interface) KM must only be used to load interface tables. Writing directly in the E-Business Suite physical tables is not supported.</p>
RKM E-Business Suite	<p>This KM reverse-engineers E-Business Suite data structures. It reverses EBS objects such as tables, views, flexfields and interface-tables structures in E-Business Suite (columns, primary keys and foreign keys).</p>

2.2 Installation and Configuration

Make sure you have read the information in this section before you start working with the E-Business Suite data:

- [System Requirements and Certifications](#)
- [Technology Specific Requirements](#)
- [Connectivity Requirements](#)

2.2.1 System Requirements and Certifications

Before performing any installation you should read the system requirements and certification documentation to ensure that your environment meets the minimum installation requirements for the products you are installing.

The list of supported platforms and versions is available on Oracle Technical Network (OTN):

<http://www.oracle.com/technology/products/oracle-data-integrator/index.html>.

2.2.2 Technology Specific Requirements

There are no technology-specific requirements for using E-Business Suite data in Oracle Data Integrator.

2.2.3 Connectivity Requirements

There are no connectivity requirements for using E-Business Suite data in Oracle Data Integrator.

2.3 Setting Up the Topology

This step consists in declaring in Oracle Data Integrator the data server, as well as the physical and logical schemas for the Oracle database that stores the E-Business Suite data.

2.3.1 Create an Oracle Data Server

Create a data server for the Oracle technology as described in "Creating an Oracle Data Server" in the *Connectivity and Knowledge Modules Guide for Oracle Data Integrator*. This data server must point to the Oracle database instance that stores the E-Business Suite data.

2.3.2 Create an Oracle Physical Schema

Create an Oracle physical schema using the standard procedure, as described in "Creating a Physical Schema" in *Administering Oracle Data Integrator*. This schema must point to the Oracle schema that contains the synonyms pointing to the E-Business Suite tables.

Note: The physical schema must represent the Oracle schema containing the synonyms pointing to the E-Business Suite tables. This schema is usually called APPS. It must not point directly to the Oracle schemas containing the Application physical tables. These are usually named after the related applications.

Note also that for reverse-engineering, the Oracle user specified in the data server to which the Physical Schema is attached, must have the privileges to select from APPLSYS tables and the Oracle Data dictionary.

Create for this physical schema a logical schema using the standard procedure, as described in "Creating a Logical Schema" in *Administering Oracle Data Integrator* and associate it in a given context.

Note: The Oracle schema containing the E-Business Suite tables and the Oracle schema containing the synonyms that point to these tables should never be defined as a Work Schema in a physical schema definition. Moreover, these Oracle schemas must not be used as staging area for a mapping.

2.4 Setting up an Integration Project

Setting up a project using E-Business Suite features follows the standard procedure. See "Creating an Integration Project" of the *Developing Integration Projects with Oracle Data Integrator*.

Import the following KMs into your Oracle Data Integrator project:

- IKM E-Business Suite (Open Interface)
- RKM E-Business Suite

In addition to these specific EBS KMs, import the standard Oracle LKMs and CKMs to perform data extraction and data quality checks with an Oracle database. See "Oracle Database" in the *Connectivity and Knowledge Modules Guide for Oracle Data Integrator* for a list of available KMs.

2.5 Creating an Oracle Model and Reverse-Engineering E-Business Suite Tables

This section contains the following topics:

- [Create an Oracle Model](#)
- [Reverse-Engineer E-Business Suite Tables](#)

2.5.1 Create an Oracle Model

Create an Oracle Model based on the Oracle technology and on the logical schema created when configuring the E-Business Suite Connection using the standard procedure, as described in "Creating a Model" of the *Developing Integration Projects with Oracle Data Integrator*.

Note: There is no E-Business Suite technology defined in Oracle Data Integrator. The data model is created on top of the logical schema corresponding to the Oracle database hosting the EBS data.

2.5.2 Reverse-Engineer E-Business Suite Tables

The RKM E-Business Suite is able to reverse-engineer the installed E-Business Suite tables, enriching them with information retrieved from the E-Business Suite Integration repository.

The reverse-engineering process returns the following information:

- The installed E-Business Suite (Modules) as sub-models
- For each module sub-model, sub-models for Tables, Views, Flexfields, and Interface Tables
- The tables as datastores with their columns as attributes, and their constraints (Primary and Foreign Keys).
- Comments on the tables

To perform a Customized Reverse-Engineering of EBS tables with the RKM E-Business Suite, use the usual procedure, as described in "Reverse-engineering a Model" of the *Developing Integration Projects with Oracle Data Integrator*. This section details only the fields specific to EBS tables:

1. In the Reverse Engineer tab of the Oracle Model, select the RKM E-Business Suite.
2. Set the RKM options as follows:
 - Applications List: Enter the list of the applications' short name, for example INV.
 - Only Installed Applications: Set this option to YES to reverse-engineer only installed and shared applications. If this option is set to NO, all applications are reverse-engineered.
 - Min Rows: Leave the default value 0, if you want to reverse-engineer all the tables. If you want to reverse-engineer only tables with a minimum number of rows, specify in this option the minimum number of rows.
 - Description Mask: Specify the description mask for filtering the reverse-engineered objects based on their description in E-Business Suite.
 - Flexfields: If this option is set to YES, applications' flexfields are reverse-engineered.
 - Interface Tables: If this option is set to YES, applications' interface tables are reverse-engineered.
3. Specify the reverse mask in the Mask field in order to select the tables to reverse. The Mask field, in the Reverse Engineer tab, filters reverse-engineered objects based on their name.

Note: The Mask field and the Description Mask option are implemented using SQL Like. The patterns that you can choose from are:

- % the percentage symbol allows you to match any string of any length (including zero length)
 - _ the underscore symbol allows you to match a single character
-

The reverse-engineering process returns the applications and tables as sub-models and datastores. You can use Oracle Applications as a source or a target of your mappings.

Features of the E-Business Suite Reverse-Engineering

Reverse-engineering E-Business Suite Tables involves the following features:

- The E-Business Suite Modules are reversed as sub-models. The sub-model names correspond to the application names.
- Each application sub-model is divided into sub-models for Tables, Views, Flexfields and Interface Tables.
- The tables/views are reversed as datastores and their columns as attributes, along with their constraints (Primary and Foreign keys).
- A sub-model called *Flexfield on <AppName>* is created for each application. Datastores in the Flexfield sub-model correspond to *Concatenated_Segment_Views* of registered Key flexfields for the application. These objects are a subset of Views. The datastores in the Flexfields sub-folder are named after the flexfields.
- Datastores in Interface-Table sub-model correspond to tables whose names contain the pattern INTERFACE. These objects are a subset of tables.

Note: Some of the Open Interfaces (as specified in EBS Integration Repository) may have interface tables whose names may not contain the pattern INTERFACE in their names.

Limitations of the E-Business Suite Reverse-Engineering Process

This section covers restrictions on reverse-engineering E-Business Suite Tables:

- Selective reverse-engineering cannot be used with this Knowledge Module.
- The Min Rows option requires Oracle statistics to be computed on all tables.
- If the Oracle user defined in the Oracle Data Integrator data server is not the owner of the tables to reverse-engineer, you must define synonyms for this user on these tables.
- Only KeyFlexfields are supported. Descriptive FlexFields are not supported.

2.6 Designing a Mapping

You can use E-Business Suite as a source and a target of a mapping.

The KM choice for a mapping determines the abilities and performance of this mapping. The recommendations in this section help in the selection of the KM for different situations concerning loading and integrating EBS data.

2.6.1 Loading Data from E-Business Suite

When using E-Business Suite as a source, you extract data from the Applications to integrate them into another system (Data warehouse, other database..).

Extracting data from E-Business Suite is performed with regular mappings sourcing from an Oracle Database. The knowledge modules working with the Oracle database technology can be used for this purpose. See "Loading Data from Oracle" in the *Connectivity and Knowledge Modules Guide for Oracle Data Integrator* for more information.

2.6.2 Integrating Data in E-Business Suite through the Open Interface

Oracle Data Integrator provides the IKM E-Business Suite (Open Interface) to integrate data in E-Business Suite. The integration process into E-Business Suite is as follows:

1. A set of Open Interface tables is loaded in a batch in a given transaction. This transaction is identified by a *Group ID*. Note the following concerning the Group ID:
 - For the first table in the batch, create a Group ID if it does not exist.
 - For the subsequent tables in the batch, use this Group ID when loading other tables in the batch.
 - When loading the last table in the batch, delete this Group ID.
2. If at any point in a batch it is required to call an E-Business Interface program, then you must validate and process data for the interface tables by executing an *Open Interface Program*. The batch is finalized by the Open Interface Program call that loads the base tables from the Open Interface tables.

These operations are supported by the IKM E-Business Suite (Open Interface). This IKM is used like the IKM Oracle Incremental Update and supports similar options to

load the Open Interface tables. This section describes the options specific to Open Interfaces. See "Oracle Database" in the *Connectivity and Knowledge Modules Guide for Oracle Data Integrator* for more information about the IKM Oracle Incremental Update.

The configuration of mappings for actions specific to E-Business Suite, such as Group ID handling and the execution of Open Interface programs, is detailed in the [Section 2.6.2.1, "Managing Group IDs"](#) and [Section 2.6.2.2, "Executing an Open Interface Program"](#).

2.6.2.1 Managing Group IDs

A transaction that integrates data into E-Business Suite is a batch identified by its *Group ID*. For example, if you load several interface tables to create a product in E-Business Suite, all of these loading operations as well as the calls to the validation and processing programs will use this batch's Group ID.

This section contains the following topics:

- [Creating a Group ID](#)
- [Using an existing Group ID](#)
- [Deleting an existing Group ID](#)

Creating a Group ID

You must force the creation of a Group ID in the first mapping that loads a group of interface tables in one single batch.

To create a Group ID in a mapping:

1. Set the following in the KM options:
 - Set `OA_CREATE_NEW_GROUP_ID` to `YES`
 - Provide a Group ID Name in the `OA_GROUP_ID_NAME` option.

Note: The Group ID Name must be unique at a given instant. You must use the `OA_REMOVE_GROUP_ID` option to remove a Group ID at the end of the batch processing.

- Give a valid SQL expression for the Group ID value in the `OA_GROUP_ID_EXPRESSION` option. Use an Oracle Database sequence value, for example `<SEQUENCE_NAME>.NEXTVAL`
2. In the mapping, select the flag `UD1` for all the columns of the interface table you wish to load with the Group ID value and set the mapping value to `0`.

In the following mappings belonging to a batch, you must use an existing Group ID.

Using an existing Group ID

To use an existing Group ID in a mapping:

1. Set `OA_USE_EXISTING_GROUP_ID` IKM option to `Yes`.
2. Provide the Group ID Name in the `OA_GROUP_ID_NAME` IKM option.
3. In the mapping, select the flag `UD1` for all the columns you wish to load with the Group ID value and set the mapping value to `0`.

In the last mapping that loads a batch of interface tables, you may delete a Group ID that is no longer necessary.

Deleting an existing Group ID

To delete an existing Group ID:

1. Select the OA_REMOVE_GROUP_ID option.
2. Provide the Group ID Name in the OA_GROUP_ID_NAME option.
3. In the mapping, select the flag UD1 for all the columns of the interface table you wish to load with the Group ID value and set the mapping value to 0.

Note: The Group IDs are stored in an SNP_OA_GROUP table that is created in the work schema specified in the physical schema that points to the Oracle Applications Interface tables. The Group ID is referenced in Oracle Data Integrator by a unique Group ID Name.

2.6.2.2 Executing an Open Interface Program

In Oracle Data Integrator mappings, when a set of interface tables is loaded, it is necessary to call an Open Interface program in order to validate and process the data in the E-Business Suite interface tables. You can use an existing Group ID in this call (see [Using an existing Group ID](#)), or create it (see [Creating a Group ID](#)) in the same mapping, if the Open Interface only contains a single table. The execution of the Open Interface program is started in the last mapping of a package. This mapping populates a set of Open Interface tables and usually deletes the Group ID, if no longer needed.

To execute an Open Interface Program:

1. Set the SUBMIT_PROGRAM option to YES.
2. Provide the name of the program to call in the OA_PROGRAM option.

Note: For a list of available Open Interface programs and their parameters, please refer to the E-Business Suite module API and Open Interface documentation or the E-Business Suite Integration repository.

3. Specify the program parameters in the OA_ARGUMENTS option. The parameters are specified in the following format:

```
argument_name => 'argument value', argument_name => 'argument value'  
...
```

If one argument must take the value of the Group ID, you must then specify
argument Name => v_group_id.

4. You must also specify the context parameters for the session that will execute the program by setting the values of the following options:
 - OA_USER_NAME: E-Business Suite User Name
 - OA_REPONSIBILITY: E-Business Suite Responsibility Name
 - OA_LANGUAGE: Language used for the responsibility
 - OA_APPLICATION: Application to which the responsibility belongs

This chapter describes how to use the knowledge modules in Oracle Data Integrator (ODI) Application Adapter for Hadoop.

It contains the following sections:

- [Introduction](#)
- [Installation and Configuration](#)
- [Setting Up the Topology](#)
- [Setting Up an Integration Project](#)
- [Creating an Oracle Data Integrator Model from a Reverse-Engineered Hive and HBase Model](#)
- [Designing a Mapping](#)

See Also: *Oracle Big Data Connectors User's Guide*

3.1 Introduction

Apache Hadoop is designed to handle and process data that is typically from data sources that are nonrelational and data volumes that are beyond what is handled by relational databases.

Oracle Data Integrator (ODI) Application Adapter for Hadoop enables data integration developers to integrate and transform data easily within Hadoop using Oracle Data Integrator. Employing familiar and easy-to-use tools and preconfigured knowledge modules (KMs), the application adapter provides the following capabilities:

- Loading data into Hadoop from the local file system, HDFS, HBase (using Hive), and SQL database (using SQOOP)
- Performing validation and transformation of data within Hadoop
- Loading processed data from Hadoop to an Oracle database, an SQL database (using SQOOP), or HBase for further processing and generating reports

Knowledge modules (KMs) contain the information needed by Oracle Data Integrator to perform a specific set of tasks against a specific technology. An application adapter is a group of knowledge modules. Thus, Oracle Data Integrator Application Adapter for Hadoop is a group of knowledge modules for accessing data stored in Hadoop.

3.1.1 Concepts

Typical processing in Hadoop includes data validation and transformations that are programmed as MapReduce jobs. Designing and implementing a MapReduce job requires expert programming knowledge. However, when you use Oracle Data Integrator and Oracle Data Integrator Application Adapter for Hadoop, you do not need to write MapReduce jobs. Oracle Data Integrator uses Apache Hive and the Hive Query Language (HiveQL), a SQL-like language for implementing MapReduce jobs.

When you implement a big data processing scenario, the first step is to load the data into Hadoop. The data source is typically in the local file system, HDFS, HBase, and Hive tables.

After the data is loaded, you can validate and transform it by using HiveQL like you use SQL. You can perform data validation (such as checking for NULLS and primary keys), and transformations (such as filtering, aggregations, set operations, and derived tables). You can also include customized procedural snippets (scripts) for processing the data.

When the data has been aggregated, condensed, or processed into a smaller data set, you can load it into an Oracle database, other relational database, or HBase for further processing and analysis. Oracle Loader for Hadoop is recommended for optimal loading into an Oracle database.

3.1.2 Knowledge Modules

Oracle Data Integrator provides the knowledge modules (KMs) described in [Table 3–1](#) for use with Hadoop.

Table 3–1 Oracle Data Integrator Application Adapter for Hadoop Knowledge Modules

KM Name	Description	Source	Target
IKM File to Hive (Load Data)	Loads data from local and HDFS files into Hive tables. It provides options for better performance through Hive partitioning and fewer data movements. This knowledge module supports wildcards (*,?).	File system	Hive
IKM Hive Control Append	Integrates data into a Hive target table in truncate/insert (append) mode. Data can be controlled (validated). Invalid data is isolated in an error table and can be recycled.	Hive	Hive
IKM Hive Transform	Integrates data into a Hive target table after the data has been transformed by a customized script such as Perl or Python	Hive	Hive
IKM File-Hive to Oracle (OLH-OSCH)	Integrates data from an HDFS file or Hive source into an Oracle database target using Oracle Loader for Hadoop, Oracle SQL Connector for HDFS, or both.	File system or Hive	Oracle Database
IKM File-Hive to SQL (SQOOP)	Integrates data from an HDFS file or Hive data source into an SQL database target using SQOOP. SQOOP uses parallel JDBC connections for loading data.	File system or Hive	SQL Database
IKM SQL to Hive-HBase-File (SQOOP)	Integrates data from an SQL database into a Hive table, HBase table, or HDFS file using SQOOP. SQOOP uses parallel JDBC connections for unloading data.	SQL Database	Hive, HBase, or File system

Table 3–1 (Cont.) Oracle Data Integrator Application Adapter for Hadoop Knowledge

KM Name	Description	Source	Target
IKM Hive to HBase Incremental Update (HBase-SerDe)	Integrates data from a Hive table into an HBase table. It supports inserting new rows and updating existing rows.	Hive	HBase
LKM HBase to Hive (HBase-SerDe)	Loads data from an HBase table into a Hive table. It provides read-only access to the source HBase table from Hive. It defines a temporary load table definition on Hive, which represents all the relevant columns of the HBase source table.	HBase	Hive
CKM Hive	Validates data against constraints	NA	Hive
RKM Hive	Reverse engineers Hive tables	Hive metadata	NA
RKM HBase	Reverse engineers HBase tables	HBase metadata	NA

3.2 Installation and Configuration

Installation requirements for Oracle Data Integrator (ODI) Application Adapter for Hadoop are provided in these topics:

- [System Requirements and Certifications](#)
- [Technology-Specific Requirements](#)
- [Location of Oracle Data Integrator Application Adapter for Hadoop](#)

3.2.1 System Requirements and Certifications

To use Oracle Data Integrator Application Adapter for Hadoop, you must first have Oracle Data Integrator, which is licensed separately from Oracle Big Data Connectors. You can download ODI from the Oracle website at

<http://www.oracle.com/technetwork/middleware/data-integrator/downloads/index.html>

Oracle Data Integrator Application Adapter for Hadoop requires a minimum version of Oracle Data Integrator 11.1.1.6.0.

Before performing any installation, read the system requirements and certification documentation to ensure that your environment meets the minimum installation requirements for the products that you are installing.

The list of supported platforms and versions is available on Oracle Technology Network:

<http://www.oracle.com/technetwork/middleware/data-integrator/overview/index.html>

3.2.2 Technology-Specific Requirements

The list of supported technologies and versions is available on Oracle Technology Network:

<http://www.oracle.com/technetwork/middleware/data-integrator/overview/index.html>

3.2.3 Location of Oracle Data Integrator Application Adapter for Hadoop

Oracle Data Integrator Application Adapter for Hadoop is available in the `ODI_HOME/odi/sdk/xml-reference` directory.

3.3 Setting Up the Topology

To set up the topology in Oracle Data Integrator, you need to identify the data server and the physical and logical schemas that store the file system, Hive, and HBase information.

This section contains the following topics:

- [Setting Up File Data Sources](#)
- [Setting Up Hive Data Sources](#)
- [Setting Up HBase Data Sources](#)
- [Connecting to a Secure Cluster](#)
- [Setting Up the Oracle Data Integrator Agent to Execute Hadoop Jobs](#)
- [Configuring Oracle Data Integrator Studio for Executing Hadoop Jobs on the Local Agent](#)

Note: Many of the environment variables described in the following sections are already configured for Oracle Big Data Appliance. See the configuration script at `/opt/oracle/odiagent-version/agent_standalone/odi/agent/bin/HadoopEnvSetup.sh`

3.3.1 Setting Up File Data Sources

In the Hadoop context, there is a distinction between files in Hadoop Distributed File System (HDFS) and local files (outside of HDFS).

To define a data source:

1. Create a Data Server object under File technology.
2. Create a Physical Schema object for every directory to be accessed.
3. Create a Logical Schema object for every directory to be accessed.
4. Create a Model for every Logical Schema.
5. Create one or more data stores for each different type of file and wildcard name pattern.
6. For HDFS files, create a Data Server object under File technology by entering the HDFS name node in the field JDBC URL and leave the JDBC Driver name empty. For example:

```
hdfs://bda1node01.example.com:8020
```

Test Connection is not supported for this Data Server configuration.

Note: No dedicated technology is defined for HDFS files.

3.3.2 Setting Up Hive Data Sources

The following steps in Oracle Data Integrator are required for connecting to a Hive system. Oracle Data Integrator connects to Hive by using JDBC.

Prerequisites

The Hive technology must be included in the standard Oracle Data Integrator technologies. If it is not, then import the technology in `INSERT_UPDATE` mode from the `xml-reference` directory.

You must add all Hive-specific flex fields.

To set up a Hive data source:

1. Create a Data Server object under Hive technology.
2. Set the following locations under JDBC:

JDBC Driver: `weblogic.jdbc.hive.HiveDriver`

JDBC URL: `jdbc:weblogic:hive://<host>:<port>[; property=value[;...]]`

For example,

```
jdbc:weblogic:hive://localhost:10000;DatabaseName=default;User=default;
Password=default
```

Note: Usually the user ID and password are provided in the respective fields of an ODI Data Server. If a Hive user is defined without a password, "password=default" is necessary as part of the URL and the password field of Data Server should be left blank.

3. Set the following under Flexfields:
 - Hive Metastore URIs: for example, `thrift://BDA:10000`
4. Ensure that the Hive server is up and running.
5. Test the connection to the Data Server.
6. Create a Physical Schema. Enter the name of the Hive schema in both schema fields of the Physical Schema definition.
7. Create a Logical Schema object.
8. Import RKM Hive into Global Objects or a project.
9. Create a new model for Hive Technology pointing to the logical schema.
10. Perform a custom reverse-engineering operation using RKM Hive.

At the end of this process, the Hive Data Model contains all Hive tables with their columns, partitioning, and clustering details stored as flex field values.

3.3.3 Setting Up HBase Data Sources

The following steps in Oracle Data Integrator are required for connecting to a HBase system.

Prerequisites

The HBase technology must be included in the standard Oracle Data Integrator technologies. If it is not, then import the technology in `INSERT_UPDATE` mode from the `xml-reference` directory.

You must add all HBase-specific flex fields.

To set up a HBase data source:

1. Create a Data Server object under HBase technology.
JDBC Driver and URL are not available for data servers of this technology.
2. Set the following under Flexfields:
HBase Quorum: Quorum of the HBase installation. For example, localhost:2181
3. Ensure that the HBase server is up and running.

Note: You cannot test the connection to the HBase Data Server.

4. Create a Physical Schema.
5. Create a Logical Schema object.
6. Import RKM HBase into Global Objects or a project.
7. Create a new model for HBase Technology pointing to the logical schema.
8. Perform a custom reverse-engineering operation using RKM HBase.

At the end of this process, the HBase Data Model contains all the HBase tables with their columns and data types.

3.3.4 Connecting to a Secure Cluster

To run the Oracle Data Integrator agent on a Hadoop cluster that is protected by Kerberos authentication, you must perform additional configuration steps.

To use a Kerberos-secured cluster:

1. Log in to the node04 of the Oracle Big Data Appliance, where the Oracle Data Integrator agent runs.
2. Generate a new Kerberos ticket for the oracle user. Use the following command, replacing realm with the actual Kerberos realm name.

```
$ kinit oracle@realm
```

3. Set the environment variables by using the following commands. Substitute the appropriate values for your appliance:

```
$ export KRB5CCNAME=Kerberos-ticket-cache-directory
```

```
$ export KRB5_CONFIG=Kerberos-configuration-file
```

```
$ export HADOOP_OPTS="$HADOOP_OPTS
```

```
-Djavax.xml.parsers.DocumentBuilderFactory=com.sun.org.apache.xerces.in  
ternal.
```

```
jaxp.DocumentBuilderFactoryImpl-Djava.security.krb5.conf=Kerberos-confi  
guration-file"
```

In this example, the configuration files are named krb5* and are located in /tmp/oracle_krb/:

```
$ export KRB5CCNAME=/tmp/oracle_krb/krb5cc_1000
```

```
$ export KRB5_CONFIG=/tmp/oracle_krb/krb5.conf
```

```
$ export HADOOP_OPTS="$HADOOP_OPTS -D
javax.xml.parsers.DocumentBuilderFactory=com.sun.org.apache.xerces.inte
rnal.jaxp.DocumentBuilderFactoryImpl -D
java.security.krb5.conf=/tmp/oracle_krb/krb5.conf"
```

4. Redefine the JDBC connection URL, using syntax like the following:

```
jdbc:hive2://node1:10000/default;principal=HiveServer2-Kerberos-Princip
al
```

For example:

```
jdbc:hive2://bdalnode01.example.com:10000/default;principal=
hive/HiveServer2Host@EXAMPLE.COM
```

See also, "HiveServer2 Security Configuration" in the CDH5 Security Guide at the following URL:

http://www.cloudera.com/content/cloudera-content/cloudera-docs/CDH5/latest/CDH5-Security-Guide/cdh5sg_hiveserver2_security.html

5. Renew the Kerberos ticket for the oracle use on a regular basis to prevent disruptions in service.

See *Oracle Big Data Appliance Software User's Guide* for instructions about managing Kerberos on Oracle Big Data Appliance.

3.3.5 Setting Up the Oracle Data Integrator Agent to Execute Hadoop Jobs

After setting up an Oracle Data Integrator agent, configure it to work with Oracle Data Integrator Application Adapter for Hadoop.

To configure the Oracle Data Integrator agent:

1. Install Hadoop on your Oracle Data Integrator agent computer.
For Oracle Big Data Appliance, see *Oracle Big Data Appliance Software User's Guide* for instructions for setting up a remote Hadoop client.
2. Install Hive on your Oracle Data Integrator agent computer.
3. Install SQOOP on your Oracle Data Integrator agent computer.
4. Set the following base environment variables for Hadoop and Hive on your ODI agent computer.

Table 3–2 Environment variables mandatory for Hadoop and Hive

Environment Variable	Value
HADOOP_HOME	Location of Hadoop dir. For example, /usr/lib/hadoop
HADOOP_CONF	Location of Hadoop configuration files such as core-default.xml, core-site.xml, and hdfs-site.xml. For example, /home/shared/hadoop-conf
HIVE_HOME	Location of Hive dir. For example, /usr/lib/hive
HIVE_CONF	Location of Hive configuration files such as hive-site.xml. For example, /home/shared/hive-conf
HADOOP_CLASSPATH	\$HIVE_HOME/lib/hive-metastore-*.jar:\$HIVE_HOME/lib/libthrift-*.jar:\$HIVE_HOME/lib/libfb*.jar:\$HIVE_HOME/lib/hive-exec-*.jar:\$HIVE_CONF
ODI_ADDITIONAL_CLASSPATH	\$HIVE_HOME/lib/'*':\$HADOOP_HOME/client/*:\$HADOOP_CONF

Table 3–2 (Cont.) Environment variables mandatory for Hadoop and Hive

Environment Variable	Value
ODI_HIVE_SESSION_JARS	<p>\$HIVE_HOME/lib/hive-contrib-*.jar:<ODI library directory>/wlhive.jar</p> <ul style="list-style-type: none"> ■ Actual path of wlhive.jar can be determined under ODI installation home. ■ Include other JAR files as required, such as custom SerDes JAR files. These JAR files are added to every Hive JDBC session and thus are added to every Hive MapReduce job. ■ List of JARs is separated by ":", wildcards in file names must not evaluate to more than one file.

- If you plan to use HBase features, set the following environment variables on your ODI agent computer. Note that you need to set these environment variables in addition to the base Hadoop and Hive environment variables.

Table 3–3 Environment Variables mandatory for HBase (In addition to base Hadoop and Hive environment variables)

Environment Variable	Value
HBASE_HOME	Location of HBase dir. For example, /usr/lib/hbase
HADOOP_CLASSPATH	\$HBASE_HOME/lib/hbase-*.jar:\$HIVE_HOME/lib/hive-hbase-handler*.jar:\$HBASE_HOME/hbase.jar
ODI_ADDITIONAL_CLASSPATH	\$HBASE_HOME/hbase.jar
ODI_HIVE_SESSION_JARS	<p>\$HBASE_HOME/hbase.jar:\$HBASE_HOME/lib/hbase-sep-api-*.jar:\$HBASE_HOME/lib/hbase-sep-impl-*hbase*.jar:/\$HBASE_HOME/lib/hbase-sep-impl-common-*.jar:/\$HBASE_HOME/lib/hbase-sep-tools-*.jar:\$HIVE_HOME/lib/hive-hbase-handler-*.jar</p>

To use Oracle Loader for Hadoop:

- Install Oracle Loader for Hadoop on your Oracle Data Integrator agent system. See *Installing Oracle Loader for Hadoop* in *Oracle Big Data Connectors User's Guide*.
- To use Oracle SQL Connector for HDFS (OLH_OUTPUT_MODE=DP_OSCH or OSCH), you must first install it. See "Oracle SQL Connector for Hadoop Distributed File System Setup" in *Oracle Big Data Connectors User's Guide*.
- Set the following environment variables for Oracle Loader for Hadoop on your ODI agent computer. Note that you must set these environment variables in addition to the base Hadoop and Hive environment variables.

Table 3–4 Environment Variables mandatory for Oracle Loader for Hadoop (In addition to base Hadoop and Hive environment variables)

Environment Variable	Value
OLH_HOME	Location of OLH installation. For example, /u01/connectors/olh
OSCH_HOME	Location of OSCH installation. For example, /u01/connectors/osch
HADOOP_CLASSPATH	<p>\$OLH_HOME/jlib/*:\$OSCH_HOME/jlib/*</p> <p>In order to work with OLH, the Hadoop jars in the HADOOP_CLASSPATH have to be manually resolved without wildcards.</p>

Table 3–4 (Cont.) Environment Variables mandatory for Oracle Loader for Hadoop (In addition to base Hadoop and Hive environment variables)

Environment Variable	Value
ODI_OLH_JARS	Comma-separated list of all JAR files required for custom input formats, Hive, Hive SerDes, and so forth, used by Oracle Loader for Hadoop. All filenames have to be expanded without wildcards. For example: \$HIVE_ HOME/lib/hive-metastore-0.10.0-cdh4.5.0.jar,\$HIVE_ HOME/lib/libthrift-0.9.0-cdh4-1.jar,\$HIVE_ HOME/lib/libfb303-0.9.0.jar
ODI_OLH_SHAREDLIBS	\$OLH_HOME/lib/libolh12.so,\$OLH_ HOME/lib/libclntsh.so.12.1,\$OLH_ HOME/lib/libnnz12.so,\$OLH_HOME/lib/libociei.so,\$OLH_ HOME/lib/libclntshcore.so.12.1,\$OLH_HOME/lib/libons.so
ODI_ADDITIONAL_CLASSPATH	\$OSCH_HOME/jlib/''

3.3.6 Configuring Oracle Data Integrator Studio for Executing Hadoop Jobs on the Local Agent

For executing Hadoop jobs on the local agent of an Oracle Data Integrator Studio installation, follow the configuration steps in the previous section with the following change: Copy JAR files into the Oracle Data Integrator userlib directory instead of the drivers directory. For example:

Linux: \$USER_HOME/.odi/oracledi/userlib directory.

Windows: C:\Users\<>USERNAME>\AppData\Roaming\odi\oracledi\userlib directory

3.4 Setting Up an Integration Project

Setting up a project follows the standard procedures. See *Developing Integration Projects with Oracle Data Integrator*.

Import the following KMs into Global Objects or a project:

- IKM File to Hive (Load Data)
- IKM Hive Control Append
- IKM Hive Transform
- IKM File-Hive to Oracle (OLH-OSCH)
- IKM File-Hive to SQL (SQOOP)
- IKM SQL to Hive-HBase-File (SQOOP)
- IKM Hive to HBase Incremental Update (HBase-SerDe)
- LKM HBase to Hive (HBase-SerDe)
- CKM Hive
- RKM Hive
- RKM HBase

3.5 Creating an Oracle Data Integrator Model from a Reverse-Engineered Hive and HBase Model

This section contains the following topics:

- [Creating a Model](#)
- [Reverse Engineering Hive Tables](#)
- [Reverse Engineering HBase Tables](#)

3.5.1 Creating a Model

To create a model that is based on the technology hosting Hive or HBase and on the logical schema created when you configured the Hive or HBase connection, follow the standard procedure described in *Developing Integration Projects with Oracle Data Integrator*.

3.5.2 Reverse Engineering Hive Tables

RKM Hive is used to reverse engineer Hive tables and views. To perform a customized reverse-engineering of Hive tables with RKM Hive, follow the usual procedures, as described in *Developing Integration Projects with Oracle Data Integrator*. This topic details information specific to Hive tables.

The reverse-engineering process creates the data stores for the corresponding Hive table or views. You can use the data stores as either a source or a target in a mapping.

RKM Hive

RKM Hive reverses these metadata elements:

- Hive tables and views as Oracle Data Integrator data stores.
Specify the reverse mask in the Mask field, and then select the tables and views to reverse. The Mask field in the Reverse Engineer tab filters reverse-engineered objects based on their names. The Mask field cannot be empty and must contain at least the percent sign (%).
- Hive columns as Oracle Data Integrator attributes with their data types.
- Information about buckets, partitioning, clusters, and sort columns are set in the respective flex fields in the Oracle Data Integrator data store or column metadata.

[Table 3–5](#) describes the created flex fields.

Table 3–5 Flex Fields for Reverse-Engineered Hive Tables and Views

Object	Flex Field Name	Flex Field Code	Flex Field Type	Description
DataStore	Hive Buckets	HIVE_BUCKETS	String	Number of buckets to be used for clustering

Table 3–5 (Cont.) Flex Fields for Reverse-Engineered Hive Tables and Views

Object	Flex Field Name	Flex Field Code	Flex Field Type	Description
Column	Hive Partition Column	HIVE_PARTITION_COLUMN	Numeric	All partitioning columns are marked as "1". Partition information can come from the following: <ul style="list-style-type: none"> ■ Mapped source column ■ Constant value specified in the target column ■ File name fragment
Column	Hive Cluster Column	HIVE_CLUSTER_COLUMN	Numeric	All cluster columns are marked as "1".
Column	Hive Sort Column	HIVE_SORT_COLUMN	Numeric	All sort columns are marked as "1".

3.5.3 Reverse Engineering HBase Tables

RKM HBase is used to reverse engineer HBase tables. To perform a customized reverse-engineering of HBase tables with RKM HBase, follow the usual procedures, as described in *Developing Integration Projects with Oracle Data Integrator*. This topic details information specific to HBase tables.

The reverse-engineering process creates the data stores for the corresponding HBase table. You can use the data stores as either a source or a target in a mapping.

RKM HBase

RKM HBase reverses these metadata elements:

- HBase tables as Oracle Data Integrator data stores.
Specify the reverse mask in the Mask field, and then select the tables to reverse. The Mask field in the Reverse Engineer tab filters reverse-engineered objects based on their names. The Mask field cannot be empty and must contain at least the percent sign (%).
- HBase columns as Oracle Data Integrator attributes with their data types.
- HBase unique row key as Oracle Data Integrator attribute called *key*.

Table 3–6 describes the options for RKM HBase.

Table 3–6 RKM HBase Options

Option	Description
SCAN_MAX_ROWS	Specifies the maximum number of rows to be scanned during reversing of a table. The default value is 10000.
SCAN_START_ROW	Specifies the key of the row to start the scan on. By default the scan will start on the first row. The row key is specified as a Java expressions returning an instance of <code>org.apache.hadoop.hbase.util.Bytes</code> . Example: <code>Bytes.toBytes(?EMP00001?)</code> .
SCAN_STOP_ROW	Specifies the key of the row to stop the scan on? By default the scan will run to the last row of the table or up to <code>SCAN_MAX_ROWS</code> is reached. The row key is specified as a Java expressions returning an instance of <code>org.apache.hadoop.hbase.util.Bytes</code> . Example: <code>Bytes.toBytes(?EMP000999?)</code> . Only applies if <code>SCAN_START_ROW</code> is specified.

Table 3–6 (Cont.) RKM HBase Options

Option	Description
SCAN_ONLY_FAMILY	Restricts the scan to column families, whose name match this pattern. SQL-LIKE wildcards percentage (%) and underscore (_) can be used. By default all column families are scanned.
LOG_FILE_NAME	Specifies the path and file name of the log file. Default path is the user home and the default file name is reverse.log.

Table 3–7 describes the created flex fields.

Table 3–7 Flex Fields for Reverse-Engineered HBase Tables

Object	Flex Field Name	Flex Field Code	Flex Field Type	Description
DataStore	HBase Quorum	HBASE_QUORUM	String	Comma separated list of Zookeeper nodes. It is used by the HBase client to locate the HBase Master server and HBase Region servers.
Column	HBase storage type	HBASE_STORAGE_TYPE	String	Defines how a data type is physically stored in HBase. Permitted values are Binary and String (default).

3.6 Designing a Mapping

After reverse engineering Hive tables and configuring them, you can choose from these mapping configurations:

- [Loading Data from Files into Hive](#)
- [Loading Data from HBase into Hive](#)
- [Loading Data from Hive into HBase](#)
- [Loading Data from an SQL Database into Hive, HBase, and File](#)
- [Validating and Transforming Data Within Hive](#)
- [Loading Data into an Oracle Database from Hive and HDFS](#)
- [Loading Data into a SQL Database from Hive and HDFS](#)

3.6.1 Loading Data from Files into Hive

To load data from the local file system or the HDFS file system into Hive tables:

1. Create the data stores for local files and HDFS files.

Refer to *Connectivity and Knowledge Modules Guide for Oracle Data Integrator* for information about reverse engineering and configuring local file data sources.
2. Create a mapping using the file data store as the source and the corresponding Hive table as the target. Use the IKM File to Hive (Load Data) knowledge module specified in the physical diagram of the mapping. This integration knowledge module loads data from flat files into Hive, replacing or appending any existing data.

IKM File to Hive

IKM File to Hive (Load Data) supports:

- One or more input files. To load multiple source files, enter an asterisk or a question mark as a wildcard character in the resource name of the file DataStore (for example, `webshop_*.log`).
- File formats:
 - Fixed length
 - Delimited
 - Customized format
- Loading options:
 - Immediate or deferred loading
 - Overwrite or append
 - Hive external tables

[Table 3–8](#) describes the options for IKM File to Hive (Load Data). See the knowledge module for additional details.

Table 3–8 *IKM File to Hive Options*

Option	Description
<code>CREATE_TARG_TABLE</code>	Check this option, if you wish to create the target table. In case <code>USE_STAGING_TABLE</code> is set to <code>false</code> , please note that data will only be read correctly, if the target table definition, particularly the row format and file format details, are correct.
<code>TRUNCATE</code>	Set this option to <code>true</code> , if you wish to replace the target table/partition content with the new data. Otherwise the new data will be appended to the target table. If <code>TRUNCATE</code> and <code>USE_STAGING_TABLE</code> are set to <code>false</code> , all source file names must be unique and must not collide with any data files already loaded into the target table.
<code>FILE_IS_LOCAL</code>	Defines whether the source file is to be considered local (outside of the current Hadoop cluster). If this option is set to <code>true</code> , the data file(s) are copied into the Hadoop cluster first. The file has to be accessible by the Hive server through the local or shared file system. If this option is set to <code>false</code> , the data file(s) are moved into the Hadoop cluster and therefore will no longer be available at their source location. If the source file is already in HDFS, setting this option is set to <code>false</code> results in just a file rename, and therefore the operation is very fast. This option only applies, if <code>EXTERNAL_TABLE</code> is set to <code>false</code> .

Table 3–8 (Cont.) IKM File to Hive Options

Option	Description
EXTERNAL_TABLE	<p>Defines whether to declare the target/staging table as externally managed. For non-external tables Hive manages all data files. That is, it will move any data files into <code><hive.metastore.warehouse.dir>/<table_name></code>. For external tables Hive does not move or delete any files. It will load data from the location given by the ODI schema.</p> <p>If this option is set to <code>true</code>:</p> <ul style="list-style-type: none"> ■ All files in the directory given by the physical data schema will be loaded. So any filename or wildcard information from the source DataStore's resource name will be ignored. ■ The directory structure and file names must comply with Hives directory organization for tables, for example, for partitioning and clustering. ■ The directory and its files must reside in HDFS. ■ No Hive LOAD-DATA-statements are submitted and thus loading of files to a specific partition (using a target-side expression) is not possible.
USE_STAGING_TABLE	<p>Defines whether an intermediate staging table will be created. A Hive staging table is required if:</p> <ul style="list-style-type: none"> ■ Target table is partitioned, but data spreads across partitions ■ Target table is clustered ■ Target table (partition) is sorted, but input file is not ■ Target table is already defined and target table definition does not match the definition required by the KM ■ Target column order does not match source file column order ■ There are any unmapped source columns ■ There are any unmapped non-partition target columns ■ The source is a fixed length file and the target has non-string columns <p>In case none of the above is <code>true</code>, this option can be turned off for better performance.</p>
DELETE_TEMPORARY_OBJECTS	<p>Removes temporary objects, such as tables, files, and scripts after integration. Set this option to <code>No</code> if you want to retain the temporary files, which might be useful for debugging.</p>

Table 3–8 (Cont.) IKM File to Hive Options

Option	Description
DEFER_TARGET_LOAD	<p>Defines whether the file(s), which have been declared to the staging table should be loaded into the target table now or during a later execution. Permitted values are <i>START</i>, <i>NEXT</i>, <i>END</i> or <i><empty></i>.</p> <p>This option only applies if <i>USE_STAGE_TABLE</i> is set to <i>true</i>.</p> <p>The typical use case for this option is when there are multiple files and each of them requires data redistribution/sorting and the files are gathered by calling the interface several times. For example, the interface is used in a package, which retrieves (many small) files from different locations and the location, stored in an Oracle Data Integrator variable, is to be used in a target partition column. In this case the first interface execution will have <i>DEFER_TARGET_LOAD</i> set to <i>START</i>, the next interface executions will have <i>DEFER_TARGET_LOAD</i> set to <i>NEXT</i> and set to <i>END</i> for the last interface. The interfaces having <i>DEFER_TARGET_LOAD</i> set to <i>START/NEXT</i> will just load the data file into HDFS (but not yet into the target table) and can be executed in parallel to accelerate file upload to cluster.</p>
OVERRIDE_ROW_FORMAT	<p>Allows to override the entire Hive row format definition of the staging table (in case <i>USE_STAGE_TABLE</i> is set to <i>true</i>) or the target table (in case <i>USE_STAGE_TABLE</i> is set to <i>false</i>). It contains the text to be used for row format definition.</p> <p>Example for reading Apache Combined WebLog files:</p> <pre>ROW FORMAT SERDE 'org.apache.hadoop.hive.contrib.serde2.RegexSerDe' WITH SERDEPROPERTIES ("input.regex" = "([^]*) ([^]*) ([^]*) (- \\ \\[[^\\]]*\\) ([^ \\"]* \"[^\"]*\") (- [0-9]*) (- [0-9]*)(\\.??\\\" \\.??\\\" \\.??\\\")\", "output.format.string" = "%1\$s %2\$s %3\$s %4\$s %5\$s %6\$s %7\$s %8\$s %9\$s %10\$s") STORED AS TEXTFILE</pre> <p>The list of columns in the source DataStore must match the list of input groups in the regular expression (same number of columns and appropriate data types). If <i>USE_STAGE_TABLE</i> is set to <i>false</i>, the number of target columns must match the number of columns returned by the SerDe, in the above example, the number of groups in the regular expression. The number of source columns is ignored (At least one column must be mapped to the target.). All source data is mapped into the target table structure according to the column order, the SerDe's first column is mapped to the first target column, the SerDe's second column is mapped to the second target column, and so on. If <i>USE_STAGE_TABLE</i> is set to <i>true</i>, the source DataStore must have as many columns as the SerDe returns columns. Only data of mapped columns will be transferred.</p>
STOP_ON_FILE_NOT_FOUND	Defines whether the KM should stop, if input file is not found.
HIVE_COMPATIBLE	<p>Specifies the Hive version compatibility. The values permitted for this option are 0.7 and 0.8.</p> <ul style="list-style-type: none"> ■ 0.7: Simulates the append behavior. Must be used for Hive 0.7 (CDH3). ■ 0.8: Uses Hive's append feature, which provides better performance. Requires Hive 0.8 (CDH4) or later.

3.6.2 Loading Data from HBase into Hive

To load data from an HBase table into Hive:

1. Create a data store for the HBase table that you want to load in Hive.
Refer to ["Setting Up HBase Data Sources"](#) for information about reverse engineering and configuring HBase data sources.
2. Create a mapping using the HBase data store as the source and the corresponding Hive table as the target. Use the LKM HBase to Hive (HBase-SerDe) knowledge module, specified in the Physical diagram of the mapping. This knowledge module provides read access to an HBase table from Hive.

LKM HBase to Hive (HBase-SerDe)

LKM HBase to Hive (HBase-SerDe) supports:

- A single source HBase table.

[Table 3–9](#) describes the options for LKM HBase to Hive (HBase-SerDe). See the knowledge module for additional details.

Table 3–9 LKM HBase to Hive (HBase-SerDe) Options

Option	Description
DELETE_TEMPORARY_OBJECTS	Deletes temporary objects such as tables, files, and scripts post data integration. Set this option to NO if you want to retain the temporary objects, which might be useful for debugging.

3.6.3 Loading Data from Hive into HBase

To load data from a Hive table into HBase:

1. Create a data store for the Hive tables that you want to load in HBase.
Refer to ["Setting Up Hive Data Sources"](#) for information about reverse engineering and configuring Hive data sources.
2. Create a mapping using the Hive data store as the source and the corresponding HBase table as the target. Use the IKM Hive to HBase Incremental Update (HBase-SerDe) knowledge module, specified in the physical diagram of the mapping. This integration knowledge module loads data from Hive into HBase. It supports inserting new rows and updating existing rows.

IKM Hive to HBase Incremental Update (HBase-SerDe)

IKM Hive to HBase Incremental Update (HBase-SerDe) supports:

- Filters, Joins, Datasets, Transformations and Aggregations in Hive
- Inline views generated by IKM Hive Transform
- Inline views generated by IKM Hive Control Append

[Table 3–10](#) describes the options for IKM Hive to HBase Incremental Update (HBase-SerDe). See the knowledge module for additional details.

Table 3–10 IKM Hive to HBase Incremental Update (HBase-SerDe) Options

Option	Description
CREATE_TARG_TABLE	Creates the HBase target table.
TRUNCATE	Replaces the target table content with the new data. If this option is set to false, the new data is appended to the target table.

Table 3–10 (Cont.) IKM Hive to HBase Incremental Update (HBase-SerDe) Options

Option	Description
DELETE_TEMPORARY_OBJECTS	Deletes temporary objects such as tables, files, and scripts post data integration. Set this option to <code>NO</code> if you want to retain the temporary objects, which might be useful for debugging.
HBASE_WAL	Enables or disables the Write-Ahead-Log (WAL) that HBase uses to protect against data loss. For better performance, WAL can be disabled.

3.6.4 Loading Data from an SQL Database into Hive, HBase, and File

To load data from an SQL Database into a Hive, HBase, and File target:

1. Create a data store for the SQL source that you want to load into Hive, HBase, or File target.

Refer to *Connectivity and Knowledge Modules Guide for Oracle Data Integrator* for information about reverse engineering and configuring SQL data sources.

2. Create a mapping using the SQL source data store as the source and the corresponding HBase table, Hive table, or HDFS files as the target. Use the IKM SQL to Hive-HBase-File (SQOOP) knowledge module, specified in the Physical diagram of the mapping. This integration knowledge module loads data from a SQL source into Hive, HBase, or Files target. It uses SQOOP to load the data into Hive, HBase, and File targets. SQOOP uses parallel JDBC connections to load the data.

IKM SQL to Hive-HBase-File (SQOOP)

IKM SQL to Hive-HBase-File (SQOOP) supports:

- Mappings on staging
- Joins on staging
- Filter expressions on staging
- Datasets
- Lookups
- Derived tables

[Table 3–11](#) describes the options for IKM SQL to Hive-HBase-File (SQOOP). See the knowledge module for additional details.

Table 3–11 IKM SQL to Hive-HBase-File (SQOOP) Options

Option	Description
CREATE_TARG_TABLE	Creates the target table. This option is applicable only if the target is Hive or HBase.
TRUNCATE	Replaces any existing target table content with the new data. For Hive and HBase targets, the target data is truncated. For File targets, the target directory is removed. For File targets, this option must be set to <code>true</code> .
SQOOP_PARALLELISM	Specifies the degree of parallelism. More precisely the number of mapper processes used for extraction. If <code>SQOOP_PARALLELISM</code> option is set to greater than 1, <code>SPLIT_BY</code> option must be defined.

Table 3–11 (Cont.) IKM SQL to Hive-HBase-File (SQOOP) Options

Option	Description
SPLIT_BY	<p>Specifies the target column to be used for splitting the source data into <i>n</i> chunks for parallel extraction, where <i>n</i> is <code>SQOOP_PARALLELISM</code>. To achieve equally sized data chunks the split column should contain homogeneously distributed values. For calculating the data chunk boundaries a query similar to <code>SELECT MIN(EMP.EMPNO), MAX(EMP.EMPNO) from EMPLOYEE EMP</code> is used. To avoid an extra full table scan the split column should be backed by an index.</p>
BOUNDARY_QUERY	<p>For splitting the source data into chunks for parallel extraction the minimum and maximum value of the split column is retrieved (KM option <code>SPLIT-BY</code>). In certain situations this may not be the best boundaries or not the most optimized way to retrieve the boundaries. In such cases this KM option can be set to a SQL query returning one row with two columns, lowest value and highest value to be used for split-column. This range will be divided into <code>SQOOP_PARALLELISM</code> chunks for parallel extraction. Example for hard-coded ranges for an Oracle source:</p> <pre>SELECT 1000, 2000 FROM DUAL</pre> <p>For preserving context independence, regular table names should be inserted through <code>odiRef.getObjectName</code> calls. For example:</p> <pre>SELECT MIN(EMPNO), MAX(EMPNO) FROM <%=odiRef.getObjectName("EMP")%></pre>
TEMP_DIR	<p>Specifies the directory used for storing temporary files, such as sqoop script, stdout and stderr redirects. Leave this option blank to use system's default temp directory:</p> <pre><?=System.getProperty("java.io.tmp")?></pre>
MAPRED_OUTPUT_BASE_DIR	<p>Specifies an hdfs directory, where <code>SQOOP</code> creates subdirectories for temporary files. A subdirectory called like the work table will be created here to hold the temporary data.</p>
DELETE_TEMPORARY_OBJECTS	<p>Deletes temporary objects such as tables, files, and scripts after data integration. Set this option to <code>NO</code> if you want to retain the temporary objects, which might be useful for debugging.</p>
USE_HIVE_STAGING_TABLE	<p>Loads data into the Hive work table before loading into the Hive target table. Set this option to <code>false</code> to load data directly into the target table.</p> <p>Setting this option to <code>false</code> is only possible, if all these conditions are true:</p> <ul style="list-style-type: none"> ■ All target columns are mapped ■ Existing Hive table uses standard hive row separators (<code>\n</code>) and column delimiter (<code>\01</code>) <p>Setting this option to <code>false</code> provides better performance by avoiding an extra data transfer step.</p> <p>This option is applicable only if the target technology is Hive.</p>
USE_GENERIC_JDBC_CONNECTOR	<p>Specifies whether to use the generic JDBC connector if a connector for the target technology is not available.</p> <p>For certain technologies <code>SQOOP</code> provides specific connectors. These connectors take care of SQL-dialects and optimize performance. When there is a connector for the respective target technology, this connector should be used. If not, the generic JDBC connector can be used.</p>
EXTRA_HADOOP_CONF_PROPERTIES	<p>Optional generic Hadoop properties.</p>

Table 3–11 (Cont.) IKM SQL to Hive-HBase-File (SQOOP) Options

Option	Description
EXTRA_SQOOP_CONF_PROPERTIES	Optional SQOOP properties.
EXTRA_SQOOP_CONNECTOR_CONF_PROPERTIES	Optional SQOOP connector properties.

3.6.5 Validating and Transforming Data Within Hive

After loading data into Hive, you can validate and transform the data using the following knowledge modules.

3.6.5.1 IKM Hive Control Append

This knowledge module validates and controls the data, and integrates it into a Hive target table in truncate/insert (append) mode. Invalid data is isolated in an error table and can be recycled. IKM Hive Control Append supports inline view mappings that use either this knowledge module or IKM Hive Transform.

[Table 3–12](#) lists the options. See the knowledge module for additional details.

Table 3–12 IKM Hive Control Append Options

Option	Description
FLOW_CONTROL	Activates flow control.
RECYCLE_ERRORS	Recycles data rejected from a previous control.
STATIC_CONTROL	Controls the target table after having inserted or updated target data.
CREATE_TARG_TABLE	Creates the target table.
TRUNCATE	Replaces the target table content with the new data. Setting this option to true provides better performance.
DELETE_TEMPORARY_OBJECTS	Removes the temporary objects, such as tables, files, and scripts after data integration. Set this option to NO if you want to retain the temporary objects, which might be useful for debugging.
HIVE_COMPATIBLE	Specifies the Hive version compatibility. The values permitted for this option are 0.7 and 0.8. <ul style="list-style-type: none"> ■ 0.7: Simulates the append behavior. Must be used for Hive 0.7 (CDH3). ■ 0.8: Uses Hive's append feature, which provides better performance. Requires Hive 0.8 (CDH4) or later.

3.6.5.2 CKM Hive

This knowledge module checks data integrity for Hive tables. It verifies the validity of the constraints of a Hive data store and diverts the invalid records to an error table. You can use CKM Hive for static control and flow control. You must also define these constraints on the stored data.

[Table 3–13](#) lists the options for this check knowledge module. See the knowledge module for additional details.

Table 3–13 CKM Hive Options

Option	Description
DROP_ERROR_TABLE	Drops error table before execution. When this option is set to YES, the error table will be dropped each time a control is performed on the target table. This means that any rejected records, identified and stored during previous control operations, will be lost. Otherwise previous rejects will be preserved. In addition to the error table, any table called <error table>_tmp will also be dropped.
HIVE_COMPATIBLE	Specifies the Hive version compatibility. The values permitted for this option are 0.7 and 0.8. <ul style="list-style-type: none"> ▪ 0.7: Simulates the append behavior. Must be used for Hive 0.7 (CDH3). ▪ 0.8: Uses Hive's append feature, which provides better performance. Requires Hive 0.8 (CDH4) or later.

3.6.5.3 IKM Hive Transform

This knowledge module performs transformations. It uses a shell script to transform the data, and then integrates it into a Hive target table using replace mode. The knowledge module supports inline view mappings and can be used as an inline-view for IKM Hive Control Append.

The transformation script must read the input columns in the order defined by the source data store. Only mapped source columns are streamed into the transformations. The transformation script must provide the output columns in the order defined by the target data store.

[Table 3–14](#) lists the options for this integration knowledge module. See the knowledge module for additional details.

Table 3–14 IKM Hive Transform Options

Option	Description
CREATE_TARG_TABLE	Creates the target table.
DELETE_TEMPORARY_OBJECTS	Removes the temporary objects, such as tables, files, and scripts post data integration. Set this option to NO if you want to retain the temporary objects, which might be useful for debugging.

Table 3–14 (Cont.) IKM Hive Transform Options

Option	Description
TRANSFORM_SCRIPT_NAME	<p>Defines the file name of the transformation script. This transformation script is used to transform the input data into the output structure. Both local and HDFS paths are supported, for example:</p> <p>Local script location: <code>file:///tmp/odi/script1.pl</code></p> <p>HDFS script location: <code>hdfs://namenode:nnPort/tmp/odi/script1.pl</code></p> <p>Ensure that the following requirements are met:</p> <ul style="list-style-type: none"> ■ The path/file must be accessible by both the ODI agent and the Hive server. Read access for the Hive server is required as it is the Hive server, which executes the resulting MR job invoking the script. ■ If TRANSFORM_SCRIPT is set (ODI creates the script file during mapping execution), the path/file must be writable for the ODI agent, as it is the ODI agent, which writes the script file using the HDFS Java API. <p>When the KM option TRANSFORM_SCRIPT is set, the following paragraphs provide some configuration help:</p> <ul style="list-style-type: none"> ■ For HDFS script locations: <p>The script file created is owned by the ODI agent user and receives the group of the owning directory. See <i>Hadoop Hdfs Permissions Guide</i> for more details. The standard configuration to cover the above two requirements for HDFS scripts is to ensure that the group of the HDFS script directory includes the ODI agent user (let's assume oracle) as well as the Hive server user (let's assume hive). Assuming that the group hadoop includes oracle and hive, the sample command below adjusts the ownership of the HDFS script directory:</p> <pre>logon as hdfs user hdfs dfs -chown oracle:hadoop /tmp/odi/myscriptdir</pre> ■ For local script locations: <p>The script file created is owned by the ODI agent user and receives the ODI agent user's default group, unless SGID has been set on the script directory. If the sticky group bit has been set, the file will be owned by the group of the script directory instead. The standard configuration to cover the above two requirements for local scripts is similar to the HDFS configuration by using the SGID:</p> <pre>chown oracle:hadoop /tmp/odi/myscriptdir chmod g+s /tmp/odi/myscriptdir</pre>
TRANSFORM_SCRIPT	<p>Defines the transformation script content. This transformation script is then used to transform the input data into the output structure. If left blank, the file given in TRANSFORM_SCRIPT_NAME must already exist. If not blank, the script file is created.</p> <p>Script example (1-to-1 transformation): <code>#!/usr/bin/csh -f</code> <code>cat</code></p> <p>All mapped source columns are spooled as tab separated data into this script via stdin. This unix script then transforms the data and writes out the data as tab separated data on stdout. The script must provide as many output columns as there are target columns.</p>

Table 3–14 (Cont.) IKM Hive Transform Options

Option	Description
TRANSFORM_SCRIPT_MODE	<p>Unix/HDFS file permissions for script file in octal notation with leading zero. For example, full permissions for owner and group: 0770.</p> <p>Warning: Using wider permissions like 0777 poses a security risk.</p> <p>See also KM option description for TRANSFORM_SCRIPT_NAME for details on directory permissions.</p>
PRE_TRANSFORM_DISTRIBUTE	Provides an optional, comma-separated list of source column names, which enables the knowledge module to distribute the data before the transformation script is applied.
PRE_TRANSFORM_SORT	Provide an optional, comma-separated list of source column names, which enables the knowledge module to sort the data before the transformation script is applied.
POST_TRANSFORM_DISTRIBUTE	Provides an optional, comma-separated list of target column names, which enables the knowledge module to distribute the data after the transformation script is applied.
POST_TRANSFORM_SORT	Provides an optional, comma-separated list of target column names, which enables the knowledge module to sort the data after the transformation script is applied.

3.6.6 Loading Data into an Oracle Database from Hive and HDFS

IKM File-Hive to Oracle (OLH-OSCH) integrates data from an HDFS file or Hive source into an Oracle database target using Oracle Loader for Hadoop. Using the mapping configuration and the selected options, the knowledge module generates an appropriate Oracle Database target instance. Hive and Hadoop versions must follow the Oracle Loader for Hadoop requirements.

See Also:

- "Oracle Loader for Hadoop Setup" in *Oracle Big Data Connectors User's Guide* for the required versions of Hadoop and Hive
- ["Setting Up the Oracle Data Integrator Agent to Execute Hadoop Jobs"](#) on page 3-7 for required environment variable settings

Table 3–15 lists the options for this integration knowledge module. See the knowledge module for additional details.

Table 3–15 IKM File - Hive to Oracle (OLH-OSCH)

Option	Description
OLH_OUTPUT_MODE	<p>Specifies how to load the Hadoop data into Oracle. Permitted values are JDBC, OCI, DP_COPY, DP_OSCH, and OSCH.</p> <ul style="list-style-type: none"> ■ JDBC output mode: The data is inserted using a number of direct insert JDBC connections. In very rare cases JDBC mode may result in duplicate records in target table due to Hadoop trying to restart tasks. ■ OCI output mode: The data is inserted using a number of direct insert OCI connections in direct path mode. If USE_ORACLE_STAGING is set to false, target table must be partitioned. If USE_ORACLE_STAGING is set to true, FLOW_TABLE_OPTIONS must explicitly specify partitioning, for example, "PARTITION BY HASH(COL1) PARTITIONS 4". In very rare cases OCI mode may result in duplicate records in target table due to Hadoop trying to restart tasks. ■ DP_COPY output mode: OLH creates a number of DataPump export files. These files are transferred by a "Hadoop fs -copyToLocal" command to the local path specified by EXT_TAB_DIR_LOCATION. Please note that the path must be accessible by the Oracle Database engine. Once the copy job is complete, an external table is defined in the target database, which accesses the files from EXT_TAB_DIR_LOCATION. ■ DP_OSCH output mode: OLH creates a number of DataPump export files. After the export phase an external table is created on the target database, which accesses these output files directly via OSCH. Please note that the path must be accessible by the Oracle Database engine. Once the copy job is complete, an external table is defined in the target database, which accesses the files from EXT_TAB_DIR_LOCATION. ■ OSCH output mode: In OSCH mode loading, OLH is bypassed. ODI creates an external table on the target database, which accesses the input files through OSCH. Please note that only delimited and fixed length files can be read. No support for loading from Hive or custom Input Formats such as RegexInputFormat, as there is no OLH pre-processing.
REJECT_LIMIT	<p>Specifies the maximum number of errors for Oracle Loader for Hadoop and external table. Examples: UNLIMITED to except all errors. Integer value (10 to allow 10 rejections) This value is used in Oracle Loader for Hadoop job definitions as well as in external table definitions.</p>
CREATE_TARG_TABLE	<p>Creates the target table.</p>
TRUNCATE	<p>Replaces the target table content with the new data.</p>
DELETE_ALL	<p>Deletes all the data in target table.</p>

Table 3–15 (Cont.) IKM File - Hive to Oracle (OLH-OSCH)

Option	Description
USE_HIVE_STAGING_TABLE	<p>Materializes Hive source data before extraction by Oracle Loader for Hadoop. If this option is set to <code>false</code>, Oracle Loader for Hadoop directly accesses the Hive source data. Setting this option to <code>false</code> is only possible, if all these conditions are true:</p> <ul style="list-style-type: none"> ■ Only a single source table ■ No transformations, filters, joins ■ No datasets <p>Setting this option to <code>false</code> provides better performance by avoiding an extra data transfer step.</p> <p>This option is applicable only if the source technology is Hive.</p>
USE_ORACLE_STAGING_TABLE	<p>Uses an intermediate Oracle database staging table.</p> <p>The extracted data is made available to Oracle by an external table. If <code>USE_ORACLE_STAGING_TABLE</code> is set to <code>true</code> (default), the external table is created as a temporary (I\$) table. This I\$ table data is then inserted into the target table. Setting this option to <code>false</code> is only possible, if all these conditions are true:</p> <ul style="list-style-type: none"> ■ <code>OLH_OUTPUT_MODE</code> is set to JDBC or OCI ■ All source columns are mapped ■ All target columns are mapped ■ No target-side mapping expressions <p>Setting this option to <code>false</code> provides better performance by avoiding an extra data transfer step, but may lead to partial data being loaded into the target table, as Oracle Loader for Hadoop loads data in multiple transactions.</p>
EXT_TAB_DIR_LOCATION	<p>Specifies the file system path of the external table. Please note the following:</p> <ul style="list-style-type: none"> ■ Only applicable, if <code>OLH_OUTPUT_MODE = DP_* OSCH</code> ■ For <code>OLH_OUTPUT_MODE = DP_*</code>: this path must be accessible both from the ODI agent and from the target database engine. ■ For <code>OLH_OUTPUT_MODE = DP_*</code>: the name of the external directory object is the I\$ table name. ■ For <code>OLH_OUTPUT_MODE = DP_COPY</code>: ODI agent will use <code>hadoop-fs</code> command to copy dp files into this directory. ■ For <code>OLH_OUTPUT_MODE = DP_* OSCH</code>: this path will contain any external table log/bad/dsc files. ■ ODI agent will remove any files from this directory during clean up before launching OLH/OSCH.
TEMP_DIR	<p>Specifies the directory used for storing temporary files, such as sqoop script, stdout and stderr redirects. Leave this option blank to use system's default temp directory:</p> <pre data-bbox="683 1644 1149 1667"><?=System.getProperty("java.io.tmp")?></pre>
MAPRED_OUTPUT_BASE_DIR	<p>Specifies an HDFS directory, where the Oracle Loader for Hadoop job will create subdirectories for temporary files/datapump output files.</p>
FLOW_TABLE_OPTIONS	<p>Specifies the attributes for the integration table at create time and used for increasing performance. This option is set by default to <code>NOLOGGING</code>. This option may be left empty.</p>

Table 3–15 (Cont.) IKM File - Hive to Oracle (OLH-OSCH)

Option	Description
DELETE_TEMPORARY_OBJECTS	Removes temporary objects, such as tables, files, and scripts post data integration. Set this option to <code>NO</code> if you want to retain the temporary objects, which might be useful for debugging.
OVERRIDE_INPUTFORMAT	<p>By default the <code>InputFormat</code> class is derived from the source <code>DataStore/Technology</code> (<code>DelimitedTextInputFormat</code> or <code>HiveToAvroInputFormat</code>). This option allows the user to specify the class name of a custom <code>InputFormat</code>. Cannot be used with <code>OLH_OUTPUT_MODE=OSCH</code>.</p> <p>Example, for reading custom file formats like web log files the OLH <code>RegexInputFormat</code> can be used by assigning the value: <code>oracle.hadoop.loader.lib.input.RegexInputFormat</code></p> <p>See KM option <code>EXTRA_OLH_CONF_PROPERTIES</code> for details on how to specify the regular expression.</p>
EXTRA_OLH_CONF_PROPERTIES	<p>Particularly when using custom <code>InputFormats</code> (see KM option <code>OVERRIDE_INPUTFORMAT</code> for details) the <code>InputFormat</code> may require additional configuration parameters. These are provided in the OLH configuration file. This KM option allows adding extra properties to the OLH configuration file. Cannot be used with <code>OLH_OUTPUT_MODE=OSCH</code>.</p> <p>Example, (loading apache weblog file format): When OLH <code>RegexInputFormat</code> is used for reading custom file formats, this KM option specifies the regular expression and other parsing details:</p> <pre><property> <name>oracle.hadoop.loader.input.regexPattern</name> <value>([]*) ([]*) ([]*) (- \\[[^\\]]*\\]) ([^ \\]* \"[^\"]*\") (- [0-9]*) (- [0-9]*) (\\.\".*?\") (\\.\".*?\") (\\.\".*?\")</value> <description>Regex for Apache WebLog format</description> </property></pre>

3.6.7 Loading Data into a SQL Database from Hive and HDFS

IKM File-Hive to SQL (SQOOP) integrates data from an HDFS file or Hive source into an SQL database target using SQOOP.

IKM File-Hive to SQL (SQOOP)

IKM File-Hive to SQL (SQOOP) supports:

- Filters, Joins, Datasets, Transformations and Aggregations in Hive
- Inline views generated by IKM Hive Control Append
- Inline views generated by IKM Hive Transform
- Hive-HBase source tables using LKM HBase to Hive (HBase SerDe)
- File source data (delimited file format only)

[Table 3–16](#) lists the options for this integration knowledge module. See the knowledge module for additional details.

Table 3–16 IKM File-Hive to SQL (SQOOP)

Option	Description
CREATE_TARG_TABLE	Creates the target table.

Table 3–16 (Cont.) IKM File-Hive to SQL (SQOOP)

Option	Description
TRUNCATE	Replaces the target datastore content with new data. If this option is set to <code>false</code> , the new data is appended to the target datastore.
DELETE_ALL	Deletes all the rows in the target datastore.
SQOOP_PARALLELISM	Specifies the degree of parallelism. More precisely the number of mappers used during SQOOP export and therefore the number of parallel JDBC connections.
USE_TARGET_STAGING_TABLE	<p>By default the source data is staged into a target-side staging table, before it is moved into the target table. If this option is set to <code>false</code>, SQOOP loads the source data directly into the target table, which provides better performance and less need for tablespace in target RDBMS by avoiding an extra data transfer step.</p> <p>For File sources setting this option to <code>false</code> is only possible, if all these conditions are met:</p> <ul style="list-style-type: none"> ■ All source columns must be mapped ■ Source and target columns have same order ■ First file column must map to first target column ■ no mapping gaps ■ only 1-to-1 mappings (no expressions) <p>Please note the following:</p> <ul style="list-style-type: none"> ■ SQOOP uses multiple writers, each having their own JDBC connection to the target. Every writer uses multiple transactions for inserting the data. This means that in case <code>USE_TARGET_STAGING_TABLE</code> is set to <code>false</code>, changes to the target table are no longer atomic and writer failures can lead to partially updated target tables. ■ The Teradata Connector for SQOOP always creates an extra staging table during load. This connector staging table is independent of the KM option.
USE_GENERIC_JDBC_CONNECTOR	<p>Specifies whether to use the generic JDBC connector if a connector for the target technology is not available.</p> <p>For certain technologies SQOOP provides specific connectors. These connectors take care of SQL-dialects and optimize performance. When there is a connector for the respective target technology, this connector should be used. If not, the generic JDBC connector can be used.</p>
FLOW_TABLE_OPTIONS	<p>When creating the target-side work table, RDBMS-specific table options can improve performance. By default this option is empty and the knowledge module will use the following table options:</p> <ul style="list-style-type: none"> ■ For Oracle: NOLOGGING ■ For DB2: NOT LOGGED INITIALLY ■ For Teradata: no fallback, no before journal, no after journal <p>Any explicit value overrides these defaults.</p>
TEMP_DIR	<p>Specifies the directory used for storing temporary files, such as sqoop script, stdout and stderr redirects. Leave this option blank to use system's default temp directory:</p> <pre><?=System.getProperty("java.io.tmp")?></pre>

Table 3–16 (Cont.) IKM File-Hive to SQL (SQOOP)

Option	Description
MAPRED_OUTPUT_BASE_DIR	Specifies an HDFS directory, where SQOOP creates subdirectories for temporary files. A subdirectory called like the work table will be created here to hold the temporary data.
DELETE_TEMPORARY_OBJECTS	Deletes temporary objects such as tables, files, and scripts after data integration. Set this option to NO if you want to retain the temporary objects, which might be useful for debugging.
TERADATA_PRIMARY_INDEX	Primary index for the target table. Teradata uses the primary index to spread data across AMPs. It is important that the chosen primary index has a high cardinality (many distinct values) to ensure evenly spread data to allow maximum processing performance. Please follow Teradata's recommendation on choosing a primary index. This option is applicable only to Teradata targets.
TERADATA_FLOW_TABLE_TYPE	Type of the Teradata flow table, either SET or MULTISSET. This option is applicable only to Teradata targets.
TERADATA_OUTPUT_METHOD	Specifies the way the Teradata Connector will load the data. Valid values are: <ul style="list-style-type: none"> ▪ batch.insert: multiple JDBC connections using batched prepared statements (simplest to start with) ▪ multiple.fastload: multiple FastLoad connections ▪ internal.fastload: single coordinated FastLoad connections (most performant) This option is applicable only to Teradata targets.
EXTRA_HADOOP_CONF_PROPERTIES	Optional generic Hadoop properties.
EXTRA_SQOOP_CONF_PROPERTIES	Optional SQOOP properties.
EXTRA_SQOOP_CONNECTOR_CONF_PROPERTIES	Optional SQOOP connector properties.

JD Edwards World

This chapter describes how to work with JD Edwards World Knowledge Modules in Oracle Data Integrator.

This chapter includes the following sections:

- [Section 4.1, "Introduction"](#)
- [Section 4.2, "Installation and Configuration"](#)
- [Section 4.3, "Setting Up the Topology"](#)
- [Section 4.4, "Setting up an Integration Project"](#)
- [Section 4.5, "Creating and Reverse-Engineering a Model"](#)
- [Section 4.6, "Designing a Mapping"](#)

4.1 Introduction

JD Edwards (JDE) World is an integrated applications suite of comprehensive ERP software that combines business value, standards-based technology, and deep industry experience into a business solution with a low total cost of ownership.

4.1.1 Concepts

The JDE Knowledge Modules for Oracle Data Integrator use mature database-level integration methods for JDE World, in order to:

- Reverse-Engineer JDE World data structures
- Read data from JDE World (Direct Database Integration)
- Write data through the Z-tables to a JDE World Application (Interface Table Integration)

4.1.2 Knowledge Modules

Oracle Data Integrator provides the Knowledge Modules listed in [Table 4-1](#) for handling JDE World data. These specific JDE World KMs provide connectivity and integration of the JDE World platform with any database application through Oracle Data Integrator.

Table 4–1 JDE Knowledge Modules

Knowledge Module	Description
RKM JDE World	Reverse-engineers the metadata of the applications' objects such as tables and interface tables from JDE World installed on DB2 iSeries database, through DB2 iSeries JDBC driver (jt400).
RKM JDE World (JDE World JDBC Driver)	Reverse-engineers the metadata of the applications' objects such as tables and interface tables from JDE World installed on DB2 iSeries database, through JDE World JDBC Driver.
IKM JDE World Control Append	Integrates data from any source to JDE World. Integrates data in a Z-table in control append mode. <ul style="list-style-type: none"> ■ Data can be controlled: invalid data is isolated in the Error Table and can be recycled ■ The KM performs integration into JDE World with a RPG program

4.2 Installation and Configuration

Make sure you have read the information in this section before you start working with the JDE World data:

- [System Requirements and Certifications](#)
- [Technology Specific Requirements](#)
- [Connectivity Requirements](#)

4.2.1 System Requirements and Certifications

Before performing any installation you should read the system requirements and certification documentation to ensure that your environment meets the minimum installation requirements for the products you are installing.

The list of supported platforms and versions is available on Oracle Technical Network (OTN):

<http://www.oracle.com/technology/products/oracle-data-integrator/index.html>

4.2.2 Technology Specific Requirements

This section lists the technology specific requirements of the JDE World Knowledge Modules.

RKM JDE World

- `jt400.jar` - This jar file must be in the `~/ .odi/oracledi/userlib` folder.

RKM JDE World (JDE World JDBC Driver)

- `JDEWorldJDBC.jar` - This JDE World JDBC driver file must be in the `~/ .odi/oracledi/userlib` folder.

IKM JDE World Control Append

The RPG program requires the following files:

- `JDEWorldJDBC.jar` - This JDE World JDBC driver file must be in the `~/ .odi/oracledi/userlib` folder.

See "Add Additional Drivers and Open Tools" in the *Installing and Configuring Oracle Data Integrator* for more information about these folders.

- BaseJar.jar - This jar file must be in the `~/.odi/oracledi/userlib` folder.
- config.xml - This configuration file must be in the `ODI_HOME/odi/studio/bin` folder.

Note: These three files are delivered with ODI and are located in the `ODI_HOME/odi/misc/jde-world` directory.

Tip: The `ODI_HOME/odi/misc/jde-world` directory also contains a `jde.properties` file. This properties file is a template that you can make use of if you are using the `JDE_SECURITY_FILE` option in the IKM JDE World Control Append. See [Section 4.6.2, "Integrating Data in JDE"](#) for more information about this option. When using this template make sure to:

- Rename the properties file
- Enter the connection information (JD Edwards World user, password, environment, and address) .
See [Table 4-2](#) for more information about the connection related options. Also, an example of a security file is provided below the table.
- Move it to a directory that is accessible only by Oracle Data Integrator Studio or the standalone agent. Ensure that this directory is not accessible to any other user as the properties file contains the user name and password.

4.2.3 Connectivity Requirements

Oracle Data Integrator connects to the database hosting the JDE World data using JDBC connectivity.

The RKM JDE World (JDE World JDBC Driver) uses the JDE World JDBC driver to access the database to extract metadata information from JDE World. The JDE World JDBC driver provides the ability for non-JDE World applications to access JDE World data while maintaining the level of security and the flexibility built into the JDE World software.

Both the IKM JDE World Control Append and the RKM JDE World use the standard IBM Toolbox for Java driver (`jt400.jar`), which runs SQL queries to insert or access the database to extract metadata information for JDE World.

Note: in the JDBC URL, use the *SQL naming convention. Do not specify the naming convention to be `system` as, for example in :
`jdbc:as400://195.10.10.13;translate
binary=true;naming=system.`

*SQL should always be used unless your application is specifically designed for *SYS. Oracle Data Integrator uses the *SQL naming convention by default.

For detailed information on JDBC connectivity with IBM DB2 for iSeries, see "IBM DB2 for iSeries Connectivity Requirements" in the *Connectivity and Knowledge Modules Guide for Oracle Data Integrator*.

4.3 Setting Up the Topology

This step consists in declaring in Oracle Data Integrator the data server, as well as the physical and logical schemas that will be used to store the JDE World data.

4.3.1 Create a Data Server

The JDE World tables are stored in an IBM DB2 for iSeries library.

When working with RKM JDE World:

Create a data server for the IBM DB2 for iSeries technology using the standard procedure, as described in "Creating a DB2/400 Data Server" of the *Connectivity and Knowledge Modules Guide for Oracle Data Integrator*:

Note: When defining the connection parameters for the data server, set in the JDBC URL field `translate binary=true`

For example:

```
jdbc:as400://10.139.142.183;translate binary=true
```

This data server must point to the library that stores the JDE World data.

When working with RKM JDE World (JDE World JDBC Driver):

Create a data server for the IBM DB2 for iSeries technology using following information:

- JDBC Driver: `com.jdedwards.as400.access.JDEWJDBCdriver`
- JDBC URL:
`jded://<host>;translatebinary=true;JDEWEnvironment=<name>;user=<name>;p
wd=<passwd>;JDEWTableNomenclature=OBJN_
OBJT;JDEWColumnNomenclature=FDFT_FDFN`

For options specified in the JDBC URL, please refer to the JDE World JDBC Driver User Guide.

4.3.2 Create a Physical Schema

Create a physical schema under the data server that you have created in [Section 4.3.1, "Create a Data Server"](#). Use the standard procedure, as described in "Creating a Physical Schema" in *Administering Oracle Data Integrator*.

This schema must point to the library that contains the JDE World tables that you want to reverse-engineer.

Note: The library storing the JDE tables should never be defined as a work schema in the physical schema definition. Moreover, this library must not be used as staging area of a mapping.

Create for this physical schema a logical schema using the standard procedure, as described in "Creating a Logical Schema" in *Administering Oracle Data Integrator* and associate it in a given context.

4.4 Setting up an Integration Project

Setting up a project using JDE World features follows the standard procedure. See "Creating an Integration Project" of the *Developing Integration Projects with Oracle Data Integrator*.

Import the following KMs into your Oracle Data Integrator project:

- IKM JDE World Control Append
- RKM JDE World
- RKM JDE World (JDE World JDBC Driver)

In addition to these specific JDE World KMs, import the standard LKMs for the technology hosting your JDE World tables. For a list of available KMs, see "IBM DB2 for iSeries Knowledge Modules" in the *Connectivity and Knowledge Modules Guide for Oracle Data Integrator*:

4.5 Creating and Reverse-Engineering a Model

This section contains the following topics:

- [Create a Model](#)
- [Reverse-Engineer JDE Tables](#)

4.5.1 Create a Model

Create a Model based on the IBM DB2/400 technology hosting the JDE World tables and on the logical schema created when configuring the JDE World connection using the standard procedure, as described in "Creating a Model" of the *Developing Integration Projects with Oracle Data Integrator*.

4.5.2 Reverse-Engineer JDE Tables

The JDE RKMs are able to reverse-engineer JDE tables. These RKMs retrieve metadata from JDE objects such as tables and interface tables.

To perform a Customized Reverse-Engineering of JDE tables with the RKM JDE World or RKM JDE World (JDE World JDBC Driver), use the usual procedure, as described in "Reverse-engineering a Model" of the *Developing Integration Projects with Oracle Data Integrator*. This section details only the fields specific to JDE World tables:

1. In the Reverse tab of the Model, select RKM JDE World or RKM JDE World (JDE World JDBC Driver).
2. Set the options for the selected RKM:

For RKM JDE World (JDE World JDBC Driver), set the options as follows:

- JDE_MODULES: Indicate the JDE System Short Name, for example:
 - 00 for Foundation Environment
 - 01 for Address Book
 - 02 for Electronic Mail.

% for all JDE Systems

Default is 01.

Note: You can also specify a list of modules. In the list, the modules must be separated by commas and enclosed within single-quote characters, for example:

'01', '02', '04'

For RKM JDE World, set the options as follows:

- JDE_DATA_TABLES: Set this option to YES to reverse-engineer data tables
- JDE_Z_TABLES: Set this option to YES to reverse-engineer interface tables (Z-tables)
- JDE_MODULES: Indicate the JDE System Short Name, for example:

00 for Foundation Environment

01 for Address Book

02 for Electronic Mail.

% for all JDE Systems

Default is 01.

Note: You can also specify a list of modules. In the list, the modules must be separated by commas and enclosed within single-quote characters, for example:

'01', '02', '04'

- JDE_LANGUAGE: Indicate the language used for retrieving object descriptions and comments, for example E for English, F for French, and S for Spanish. Default is E.
3. Specify the reverse mask in the Mask field in order to select the tables to reverse. The Mask field, in the Reverse tab, filters reverse-engineered objects based on their name. The Mask field must not be empty and must contain at least the percentage symbol (%).

The reverse-engineering process returns the datastores grouped per module. You can use these datastores as a source or a target of your mappings.

4.6 Designing a Mapping

You can use JDE World data tables as a source of a mapping. JDE Z-tables can be used as the target of a mapping.

The KM choice for a mapping determines the abilities and performance of this mapping. The recommendations in this section help in the selection of the KM for different situations concerning loading and integrating JDE World data.

4.6.1 Loading Data from JDE

After performing a reverse-engineering using the RKM JDE World or RKM JDE World (JDE World JDBC Driver), you can use JDE World data tables as a source of a mapping to extract data from the JDE World application and integrate them into another system (Data warehouse, other database and so forth).

Using JDE World as a source in these conditions is the same as using a DB2/400 as a source in a mapping. The generic SQL and IBM DB2 for iSeries KMs can be used for this purpose. See the following chapters in the *Connectivity and Knowledge Modules Guide for Oracle Data Integrator* for more information:

- "Generic SQL"
- "IBM DB2 for iSeries"

4.6.2 Integrating Data in JDE

After performing a reverse-engineering using the RKM JDE World, you can use JDE Z-tables as a target of a mapping to load data from any system to the JDE World application with the IKM JDE World Control Append.

The integration of data into JDE World is performed in two phases:

- During the first phase data is integrated into a set of Z-tables using several mappings, without calling the RPG program. These mappings can use the IKM JDE World Control Append with the JDE_INVOKE option set to No.
- During the second phase the RPG program is launched to integrate the data from these Z-tables into JDE World. This is typically done in the mapping loading the last required Z-table. This mapping also uses the IKM JDE World Control Append with the JDE_INVOKE option set to Yes.

These mappings should be sequenced in a package.

Oracle Data Integrator can automatically call the RPG program to write to JDE World. The RPG program call should be activated in the IKM only after loading all the required Z-table for populating JDE. The capability to load the Z-Tables, as well as the call of the RPG program is provided by the IKM JDE World Control Append.

To create a mapping targeting JDE World:

1. Create a mapping with Z-tables as target datastores.
2. Create joins, filters, and mappings as usual.
3. In the Physical diagram properties, go to the Integration Knowledge Module tab and select the **IKM JDE World Control Append**.
4. Set the standard KM options (INSERT, COMMIT, FLOW_CONTROL).
5. If this mapping launches the RPG program, specify the KM options as follows:
 1. Set the JDE_INVOKE option to Yes.
 2. If you want to create a security file, set the connection related options as shown in [Table 4-2](#).

Table 4–2 Connection Related KM Options

Option	Value	Notes
USE_SECURITY_FILE	Yes	To enhance security when the RPG program is submitted, the system reads the text file specified in the JDE_SECURITY_FILE option and uses the JD Edwards World user, password, environment, and address as indicated in the text file.
JDE_SECURITY_FILE	Absolute path of the connection security file	This file contains the JDE World user, password, environment, and address specified in the JDE_USER, JDE_PASSWORD, JDE_ENVIRONMENT, and JDE_ADDRESS options. See the security file example below for more information. Mandatory if USE_SECURITY_FILE is set to yes. In this case, there is no need to set valuse for JDE_USER, JDE_PASSWORD, JDE_ENVIRONMENT, and JDE_ADDRESS. You can use the template security file that is delivered with the files required for the RPG program. For more information, see the Tip in Section 4.2.2, "Technology Specific Requirements" .
JDE_USER	JDE World user	The user must have the ONEWORLD profile.
JDE_PASSWORD	JDE World password	This user's JDE World password.
JDE_ENVIRONMENT	JDE World environment	The JDE World environment
JDE_ADDRESS	JDE World address	The IP or full address of the iSeries server

The following example shows a security file. Make sure to use the same syntax and key names in your security file.

```
Username=ODI
Password=password
Environment=JDEENV
Address=iseries.organization.com
```

3. Set the parameters for the RPG program as shown in [Table 4–3](#).

Table 4–3 RPG Program related KM Options

Option	Value	Notes
JDE_PRGNAME	The name of the PRG program	For example: P01051Z
JDE_CLNAME	The type of the PRG program	For example: J01051Z
JDE_VERSION	The name of the version of the report that you want to process	For example: XJDE0006. Note: <ul style="list-style-type: none"> ■ Enter the version name of the report to duplicate and process; you cannot submit the template of a report ■ ODI creates a temporary version

Table 4–3 (Cont.) RPG Program related KM Options

Option	Value	Notes
JDE_KCO	The batch ID	For example: 1 Note that the batch ID allows to specify which rows are to be processed by the RPG programm.
JDE_EDTN	The transaction ID	For example: 1 Note that transaction ID allows to group several rows in a single transaction. This is typically the case for header-detail tables. The headers are linked to the detail by the transaction id. This IKM does not provide truncation.

Limitations of the IKM JDE World Control Append

- When using the RECYCLE_ERRORS option, an Update Key must be set for your mapping.
- When using this module with a journalized source table, data is automatically filtered to not include source deletions.
- The FLOW_CONTROL and STATIC_CONTROL options call the Check Knowledge Module to isolate invalid data (if no CKM is set, an error occurs).
- The RPG program must be executed on the JDE World iSeries server.
- The Oracle Data Integrator run-time agent can be installed on this server. However, it is not necessary to install the run-time agent on the server to run the RPG program. The RPG program can be executed using a local agent.
- Besides the information whether the RPG program has been started or not, ODI does not give any further details about the execution of the program. To know more about the execution of the program you can view the log file created by the iSeries server and issue the Work with Spooled Files (WRKSPLF) command.

This chapter describes how to work with Oracle PeopleSoft Knowledge Modules in Oracle Data Integrator.

This chapter includes the following sections:

- [Section 5.1, "Introduction"](#)
- [Section 5.2, "Installation and Configuration"](#)
- [Section 5.3, "Setting up the Topology"](#)
- [Section 5.4, "Setting up the Project"](#)
- [Section 5.5, "Creating and Reverse-Engineering a Model"](#)
- [Section 5.6, "Designing a Mapping"](#)

5.1 Introduction

Oracle Data Integrator integrates data extracted from Oracle PeopleSoft applications. It supports reverse-engineering of PeopleSoft metadata as well as scalable data extraction from PeopleSoft tables.

5.1.1 Concepts

The Oracle Data Integrator KMs for PeopleSoft use mature integration methods for PeopleSoft, in order to:

- Reverse-Engineer PeopleSoft data structures (Business Objects, tables, views, columns, keys, and foreign keys) in the form of datastores. Oracle Data Integrator provides two specialized knowledge modules (KMs) for performing this operation for PeopleSoft instances hosted in Oracle and Microsoft SQL Server databases.
- Extract data from PeopleSoft using a data-level integration approach. Data is extracted from the database tables containing the PeopleSoft data.

5.1.2 Knowledge Modules

Oracle Data Integrator provides the Knowledge Modules listed in [Table 5-1](#) for handling PeopleSoft data. These specific Knowledge Modules for PeopleSoft provide integration and connectivity between Oracle Data Integrator and the PeopleSoft platform.

These KMs enable data-level integration for PeopleSoft: Data extraction is performed directly on the PeopleSoft Business Objects tables. This method is read-only.

Table 5–1 Oracle PeopleSoft Knowledge Modules

Knowledge Module	Description
RKM PeopleSoft ORACLE	Reverse-engineering knowledge module to retrieve the business objects, tables, views, columns, keys, and foreign keys from PeopleSoft. The database hosting the PeopleSoft tables is Oracle.

5.2 Installation and Configuration

Make sure you have read the information in this section before you start working with the PeopleSoft technology:

- [System Requirements and Certifications](#)
- [Technology Specific Requirements](#)
- [Connectivity Requirements](#)

5.2.1 System Requirements and Certifications

Before performing any installation you should read the system requirements and certification documentation to ensure that your environment meets the minimum installation requirements for the products you are installing.

The list of supported platforms and versions is available on Oracle Technical Network (OTN):

<http://www.oracle.com/technology/products/oracle-data-integrator/index.html>

5.2.2 Technology Specific Requirements

There are no technology-specific requirements for using PeopleSoft data in Oracle Data Integrator.

5.2.3 Connectivity Requirements

Oracle Data Integrator connects the database hosting the PeopleSoft data using JDBC connectivity. For detailed information on JDBC connectivity with Oracle and Microsoft SQL Server databases, see the "Oracle Database Connectivity Requirements" and "Microsoft SQL Server Connectivity Requirements" in the *Oracle Fusion Middleware Connectivity and Knowledge Modules Guide for Oracle Data Integrator*.

5.3 Setting up the Topology

This step consists in declaring in Oracle Data Integrator the data server, as well as the physical and logical schemas that will be used to store the PeopleSoft data.

5.3.1 Create a Data Server

The PeopleSoft tables can be stored in an Oracle schema or a Microsoft SQL Server database.

Create a data server either for the Oracle technology or for the Microsoft SQL Server technology. See "Creating an Oracle Data Server" or "Creating a Microsoft SQL Server Data Server" in the *Oracle Fusion Middleware Connectivity and Knowledge Modules Guide for Oracle Data Integrator*.

This data server represents the database instance that stores the PeopleSoft data.

5.3.2 Create a Physical Schema

Create a physical schema under the data server that you have created in [Section 5.3.1, "Create a Data Server"](#). Use the standard procedure, as described in "Creating a Physical Schema" in *Administering Oracle Data Integrator*.

This schema is the Oracle schema or the Microsoft SQL Server database that contains the PeopleSoft tables you want to reverse-engineer.

Note: The Oracle schema or the Microsoft SQL Server database storing the PeopleSoft tables should not be defined as a work schema in the physical schema definition. Moreover, this schema or database must not be used as staging area for a mapping.

Create for this physical schema a logical schema using the standard procedure, as described in "Creating a Logical Schema" in *Administering Oracle Data Integrator* and associate it in a given context.

5.4 Setting up the Project

Setting up a project using PeopleSoft features follows the standard procedure. See "Creating an Integration Project" of the *Oracle Fusion Middleware Developer's Guide for Oracle Data Integrator*.

Import the following KMs into your Oracle Data Integrator project:

- RKM PeopleSoft ORACLE

In addition to these specific PeopleSoft KMs, import the standard LKMs for the Oracle or Microsoft SQL Server technologies. See Oracle Database "Knowledge Modules" and Microsoft SQL Server "Knowledge Modules" in the *Connectivity and Knowledge Modules Guide for Oracle Data Integrator* for a list of available KMs.

5.5 Creating and Reverse-Engineering a Model

This section contains the following topics:

- [Create a Model](#)
- [Reverse-Engineer PeopleSoft Tables](#)

5.5.1 Create a Model

Create a Model based on the Oracle or Microsoft SQL technology and on the logical schema created when setting up the topology using the standard procedure, as described in "Creating a Model" of the *Oracle Fusion Middleware Developer's Guide for Oracle Data Integrator*.

5.5.2 Reverse-Engineer PeopleSoft Tables

The PeopleSoft RKMs are able to reverse-engineer PeopleSoft data structures at data level, enriching them with information retrieved from the PeopleSoft dictionary.

Data extraction is performed directly on the PeopleSoft Business Objects tables. This access method is read-only.

The reverse-engineering process returns the following information:

- Business Objects as sub-models
- Business Objects tables as datastores with their associated columns as attributes and constraints
- Comments attached to the reversed tables and columns

To perform a Customized Reverse-Engineering of PeopleSoft tables with the PeopleSoft RKM, use the usual procedure, as described in "Reverse-engineering a Model" of the *Oracle Fusion Middleware Developer's Guide for Oracle Data Integrator*. This section details only the fields specific to PeopleSoft tables:

1. In the Reverse tab of the Model, select the RKM PeopleSoft ORACLE.
2. Set the BUSINESS OBJECT RKM option as follows:

Enter the Business Object code, for example CCM, DBI, or DPO.

The Business Object code corresponds to the PeopleSoft *Object Owner ID*. The different Object Owner IDs are listed in the PeopleSoft view: `EO_BCOWNRID_VW`. This field is used as a mask to filter the Business Objects to be reverse-engineered. This field must not be empty and must contain at least the percentage symbol (%).

3. Specify the reverse-engineering mask in the Mask field in order to select the tables to reverse. The Mask field, in the Reverse tab, filters reverse-engineered objects based on their name. The Mask field must not be empty and must contain at least the percentage symbol (%).

The reverse-engineering process returns the applications and tables as sub-models and datastores. You can use these PeopleSoft datastores as a source in a mapping.

5.6 Designing a Mapping

You can use PeopleSoft data tables as a source of a mapping.

The KM choice for a mapping determines the abilities and performance of this mapping. The recommendations in this section help in the selection of the KM for different situations concerning loading PeopleSoft data.

5.6.1 Loading Data from PeopleSoft

PeopleSoft data tables can be used as a source of a mapping to extract data from the PeopleSoft database and integrate them into another system (Data warehouse, other database...).

Using PeopleSoft as a source in these conditions is identical to using an Oracle or Microsoft SQL Server datastore as a source in a mapping. The generic SQL, Oracle Database, or Microsoft SQL Server KMs can be used for this purpose. See the following chapters in the *Oracle Fusion Middleware Connectivity and Knowledge Modules Guide for Oracle Data Integrator* for more information:

- "Oracle Database"
- "Generic SQL"
- "Microsoft SQL Server"

This chapter describes how to work with SAP BW Knowledge Modules in Oracle Data Integrator.

This chapter includes the following sections:

- [Section 6.1, "Introduction"](#)
- [Section 6.2, "Installation and Configuration"](#)
- [Section 6.3, "Defining the Topology"](#)
- [Section 6.4, "Setting up the Project"](#)
- [Section 6.5, "Creating and Reverse-Engineering a Model"](#)
- [Section 6.6, "Designing a Mapping"](#)
- [Section 6.7, "Considerations for SAP BW Integration"](#)

6.1 Introduction

The SAP BW Knowledge Modules let Oracle Data Integrator connect to SAP-BW system using SAP Java Connector (SAP JCo) libraries. These adapters allow mass data extraction from SAP-BW systems.

If this is the first time you are using the SAP BW adapter, it is recommended to review *Getting Started with SAP ABAP BW Adapter for Oracle Data Integrator*.

It contains the complete pre-requisites list as well as step-by-step instructions including SAP connection testing.

6.1.1 Concepts

The SAP BW Knowledge Modules for Oracle Data Integrator use mature integration methods for SAP-BW system, in order to:

- Reverse-Engineer SAP BW metadata
- Extract and load data from SAP BW system (source) to an Oracle or non-Oracle Staging Area

The reverse-engineering process returns the following SAP BW objects inside an ODI model:

- Each ODS/DSO object is represented as an ODI datastore.
- Each InfoObject will be represented in ODI as a submodel containing up to three datastores:

- InfoObjects having master data have a master data datastore containing all InfoObject attributes
- InfoObjects having attached text data have a text datastore containing all text related attributes
- InfoObjects having hierarchies defined have a hierarchy datastore containing all hierarchy related attributes
- Each InfoCube will be represented as a single ODI datastore. This datastore includes attributes for all characteristics of all dimensions as well as for all key figures.
- Each OpenHubDestination is represented as an ODI datastore.

6.1.2 Knowledge Modules

Oracle Data Integrator provides the Knowledge Modules listed in [Table 6–1](#) for handling SAP BW data.

The Oracle Data Integrator SAP BW Knowledge Modules provide integration from SAP BW systems using SAP JCo libraries. This set of KMs has the following features:

- Reads SAP BW data from SAP BW system.
- Loads this data into Oracle or non-Oracle Staging Area.
- Reverse-engineers SAP Metadata and proposes a tree browser to select only the required Metadata.
- Uses flexfields to map the SAP BW data targets types (InfoCube, InfoObject, ODS/DSO, OpenHub and Text Table) and their attributes.

Table 6–1 SAP BW Knowledge Modules

Knowledge Module	Description
LKM SAP BW to Oracle (SQLLDR)	Extracts data from SAP BW system into a flat file and then loads it into Oracle Staging Area using the SQL*LOADER command line utility.
RKM SAP ERP Connection Test	This RKM is used for testing the SAP connection from Oracle Data Integrator. See Appendix B, "Additional Information for SAP ABAP ERP Adapter" for more information.
RKM SAP BW	Reverse-engineering Knowledge Module to retrieve SAP specific metadata for InfoCubes, InfoObjects (including Texts and Hierarchies), ODS/DSO and OpenHubDestinations.
LKM SAP BW to SQL	Extracts data from SAP BW into a flat file and then loads it into a Staging Area using a JDBC connection.

6.1.3 Overview of the SAP BW Integration Process

The RKM SAP BW enables Oracle Data Integrator (ODI) to connect to SAP BW system using SAP JCo libraries and perform a customized reverse-engineering of SAP BW metadata.

The LKM SAP BW to Oracle (SQLLDR) and LKM SAP BW to SQL are in charge of extracting and loading data from SAP BW system (source) to an Oracle or non-Oracle Staging Area.

Note: Access to SAP BW is made through ABAP. As a consequence, the technology used for connecting is SAP ABAP, and the topology elements, as well as the model will be based on the SAP ABAP technology. There is no SAP BW technology in ODI, but SAP BW-specific KMs based on the SAP ABAP technology.

6.1.3.1 Reverse-Engineering Process

Reverse-engineering uses the RKM SAP BW.

This knowledge module automatically installs dedicated RFC programs to retrieve SAP BW metadata. It extracts the list of SAP BW data objects and optionally displays this list in a Metadata Browser graphical interface. The user selects from this list the SAP BW objects to reverse-engineer.

In the reverse-engineering process, data targets, primary keys, foreign keys and index are reverse-engineered into an Oracle Data Integrator model.

6.1.3.2 Integration Process

Data integration from SAP is managed by the LKM SAP BW to Oracle (SQLLDR) and the LKM SAP BW to SQL.

The LKM SAP BW to Oracle (SQLLDR) is used for mappings sourcing from SAP via ABAP and having a Staging Area located in an Oracle Database and the LKM SAP BW to SQL is used for non-Oracle staging areas.

The KM first generates optimized ABAP code corresponding to the extraction process required for a given mapping. This code includes filters and joins that can be processed directly in the source SAP BW server. This ABAP program is automatically uploaded and is executed using the OdiSAPAbapExecute tool to generate an extraction file in SAP.

The KM then transfers this extraction file either to a pre-configured FTP server or to a shared directory. This file is then either downloaded from this server using FTP, SFTP, SCP or copied to the machine where the ODI Agent is located, and is finally loaded either using SQL*Loader or using a JDBC connection to the staging area. The agent can also directly read the extraction file on the FTP server's disk. See [Section 6.7.1, "File Transfer Configurations"](#) for more information.

The rest of the integration process (data integrity check and integration) is managed with other Oracle Data Integration KMs.

6.2 Installation and Configuration

Make sure you have read the information in this section before you start working with the SAP BW data:

- [System Requirements and Certifications](#)
- [Technology Specific Requirements](#)
- [Connectivity Requirements](#)

6.2.1 System Requirements and Certifications

Before performing any installation you should read the system requirements and certification documentation to ensure that your environment meets the minimum installation requirements for the products you are installing.

The list of supported platforms and versions is available on Oracle Technical Network (OTN):

<http://www.oracle.com/technology/products/oracle-data-integrator/index.html>.

6.2.2 Technology Specific Requirements

Some of the Knowledge Modules for SAP BW use specific features of SAP-BW system and Oracle database. This section lists the requirements related to these features.

- A JCo version compatible with adapter must be used. The list of supported JCo versions is available from the Oracle Technology Network (OTN). See [Section 6.2.1, "System Requirements and Certifications"](#) for more information.
- A JVM version compatible with both Oracle Data Integrator and JCo must be used.
- The adapter supports two transfer modes for transferring data from SAP system to the ODI agent: data transfer using a Shared Directory and data transfer through FTP. For details and restrictions, see [Section 6.7.1, "File Transfer Configurations"](#).

Depending on the chosen file transfer mode the following requirements must be met:

– **Data transfer through a Shared Directory (recommended transfer method)**

The LKM SAP BW to Oracle (SQLLDR) requires a folder that is shared between the SAP system and the ODI agent. The SAP application server transfers the data by writing it out into a folder that is accessible from the SAP system and the ODI agent machine. This is typically done by sharing a folder of the ODI agent machine with the SAP system. Note that the shared folder does not necessarily have to be located on the ODI agent machine. A shared folder on a third machine is also possible, as long as the shared folder is accessible to both the ODI agent machine and the SAP system.

Note: For security reasons, folders located on the SAP server should not be shared. You should instead share a folder located of the ODI agent machine with the SAP system, or use a third machine as the shared file server.

The shared folder must be accessible to SAP system and not just to the underlying operating system. This means that the folder needs to be declared in SAP transaction AL11 and the folder opens successfully in AL11.

– **Data transfer through FTP**

LKM SAP BW to Oracle (SQLLDR) requires a FTP server to upload data from the SAP BW system. This data is either read locally by the agent executing the mapping (when this agent runs on the FTP server machine), or remotely (when this agent is located on a different machine than the FTP server). This FTP server must be accessible over the network from both the SAP BW machine and the agent machine.

- For "LKM SAP BW to Oracle (SQLLDR)" only: SQL*Loader is required on the machine running the agent when executing mappings using LKM SAP BW to Oracle (SQLLDR). SQL*Loader is used for loading data extracted from SAP to the Oracle staging area.

6.2.3 Connectivity Requirements

Oracle Data Integrator connects to the SAP BW system hosting the SAP BW data using JCo. It also uses a FTP Server or a shared directory to host the data extracted from the SAP system.

This section describes the required connection information:

- [Installing and Configuring JCo](#)
- [Installing ODI SAP Components into SAP System](#)
- [Validating the SAP Environment Setup](#)
- [Gathering SAP Connection Information](#)
- [Gathering FTP Connection Information](#)
- [Gathering Shared Directory Information](#)
- [Adding the Open Tool](#)

6.2.3.1 Installing and Configuring JCo

The SAP adapter uses JCo to connect to the SAP system. JCo must be configured before proceeding with the project.

To install and configure JCo:

1. Download a supported JCo version for your configuration from <http://service.sap.com/connectors>. Check the supported JCo version in the Compatibility Matrix available at Oracle Technology Network:
http://www.oracle.com/technology/products/oracle-data-integrator/10.1.3/htdocs/documentation/odi_certification.xls

Notes:

- A minimum version of JCo 3.0.2 is required
 - Choose the SAP JCo package matching your operating system and your system architecture (32/64Bit). E.g. if you are running ODI inside a 32-Bit JVM, you must download the 32-Bit SAP JCo, even if the CPU and OS are 64-Bit. Mixing 32-bit and 64-bit architecture is not possible due to native libraries required by SAP JCo and will result in connection failure.
 - `odi.conf` contains the JDK path used for ODI Studio.
-
-
2. Unzip the appropriate distribution package into an arbitrary directory `<sapjco-install-path>`.
 3. Follow the installation instructions provided in the JCo documentation (`<sapjco-install-path>/javadoc/installation.html`) for your platform.
 4. Copy the required files (`sapjco3.jar` and `sapjco3.dll/.so`) into the `<ODI_HOME>/odi/sdk/lib` directory (ODI Studio, ODI Standalone Agent) and into the `<WLS_DOMAIN>/lib` directory (JEE Agent).
 5. Restart the ODI Components using SAP (ODI Studio, Standalone Agent)
 6. Check the JCo installation.

Note: For SAP Secure Network Connections, please ensure that:

- the environment variable SECUDIR points to the directory containing any certificate files.
- the environment variable SNC_LIB points to the directory containing sapcrypto.dll/libsapcrypto.so for Linux.

Please contact your SAP Administrator for more details on certificates and Crypto Libraries.

6.2.3.2 Installing ODI SAP Components into SAP System

The ODI SAP adapter communicates with the SAP System using a few ODI SAP RFCs. These RFCs are installed by your SAP Basis team using SAP Transport requests. Please contact your SAP administrators for installing the ODI SAP Components and assigning the required SAP user authorizations following the instructions given in [Appendix C.2, "Installing ODI SAP Components"](#).

6.2.3.3 Validating the SAP Environment Setup

[Appendix C.3, "Validating the ODI SAP Setup"](#) contains instructions for some basic validation of the SAP environment for the use with the ODI SAP Adapter. Please ask your SAP Basis team to run all validations and provide back the validation results like screen shots and confirmations.

6.2.3.4 Gathering SAP Connection Information

In order to connect to the SAP BW system, you must request the following information from your SAP administrators:

- for SAP Group Logons:
 - SAP System ID: The three-character, unique identifier of a SAP system in a landscape.
 - Message Server: IP address/ host name of SAP Message Server.
 - Message Server Port: port number or service name of SAP Message Server.
 - Group/Server: Name of SAP Logon Group.
- for SAP Server Logons:
 - SAP BW System IP Address or Hostname: IP address/ host name of the host on which SAP is running.
 - SAP Client Number: The three-digit number assigned to the self-contained unit which is called Client in SAP. A Client can be a training, development, testing or production client or represent different divisions in a large company.
 - SAP System Number: The two-digit number assigned to a SAP instance which is also called Web Application Server or WAS.
 - SAP System ID: The three-character, unique identifier of a SAP system in a landscape.
- SAP BW System IP Address or Hostname: IP address/ host name of the host on which SAP is running.
- SAP User: SAP User is the unique user name given to a user for logging on the SAP System.

- SAP Password: Case-sensitive password used by the user to log in.
- SAP Language: Code of the language used when logging in. For example: EN for English, DE for German.
- SAP Client Number: The three-digit number assigned to the self-contained unit which is called Client in SAP. A Client can be a training, development, testing or production client or represent different divisions in a large company.
- SAP System Number: The two-digit number assigned to a SAP instance which is also called Web Application Server or WAS.
- SAP System ID: The three-character, unique identifier of a SAP system in a landscape.
- SAP SNC Connection Properties (optional) SAP Router String (optional): SAP is enhancing security through SNC and SAP router. It is used when these securities are implemented.
- SAP Transport Layer Name: This string uniquely identifies a transport layer in a SAP landscape. It allows ODI to create transport requests for later deployment in SAP. Even though there is a default value here, this transport layer name *must* be provided by your SAP Basis team. Not doing so may result in significant delays during installation.
- SAP Temporary Directory (`SAP_TMP_DIR`): This parameter is used to define the custom work directory on the SAP system. It applies only in case of FTP transfer mode. This path is used for temporary files during extraction. The default value (FlexField value is left empty) is to use the `DIR_HOME` path defined in the SAP profile.

Specify a directory on the SAP application server to be used for temporary files. The path must end on slash or backslash depending on the OS the SAP application server runs on. The ODI SAP user must have read and write privileges on this directory.

Caution: The default value (empty FF value) must be avoided for any critical SAP system, as an excessive temp file during extraction could fill up the respective file system and can cause serious issues on the SAP system. Even for SAP development systems it is strongly recommended to override the default value.

- SAP BW Version: 3.5 or 7.0
- SAP Character Set: The character set is only required if your SAP system is not a UNICODE system. For a complete list of character sets, see "Locale Data" in the *Oracle Database Globalization Support Guide*. For example, EE8ISO8859P2 for Croatian Data. For UNICODE systems, use UTF8.

Note: All the connection data listed above (except SAP SNC Connection Properties and SAP Router String) are mandatory and should be requested from the SAP Administrators. You may consider requesting support during connection setup from your SAP administrators.

6.2.3.5 Gathering FTP Connection Information

The SAP BW system will push data to a server using the FTP protocol. Collect the following information from your system administrator:

- FTP server name or IP address
- FTP login ID
- FTP login password
- Directory path for storing temporary data files

Validate that the FTP server is accessible both from SAP System and from ODI agent machine.

6.2.3.6 Gathering Shared Directory Information

Gathering Shared Directory information only applies, if you plan to transfer data through a shared directory. The SAP system will push data to a shared folder. For later setup, gather the following information from your system administrator:

- (UNC) path name of the shared folder

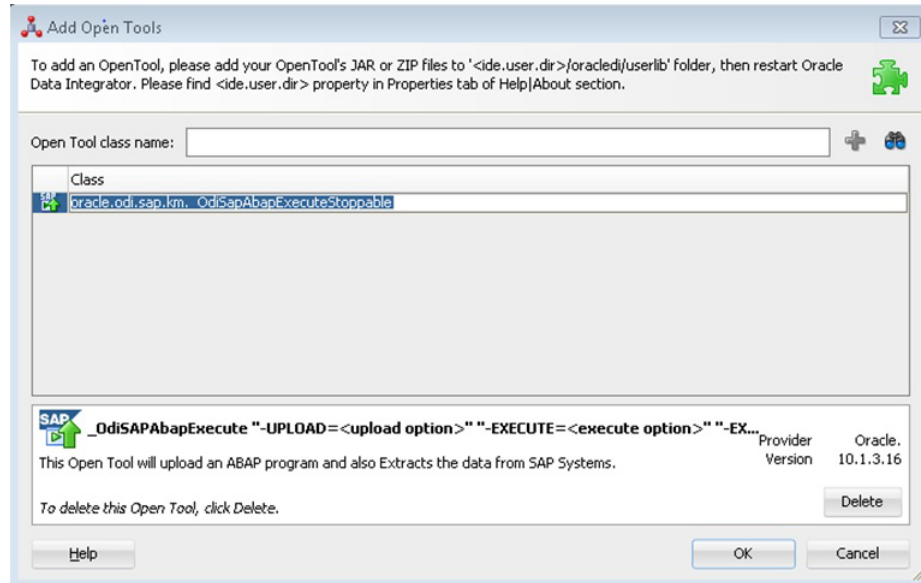
Validate that the shared folder is accessible from both the SAP System and the ODI agent machine and does not require any interactive authentication to be accessed.

Please note that the shared folder must be accessible from the SAP system using the `<sid>adm` user and from the operating system user that starts the ODI agent.

6.2.4 Adding the Open Tool

The complete process of installing and adding an Open Tool to ODI is described in *Oracle Data Integrator Tool Reference*. This section details only the SAP ABAP specific steps.

1. Connect to Designer.
2. Select **File > Add/Remove Open Tools...**
3. In the Add/remove Open Tools window, enter the following name in the Open Tool class name field:
`oracle.odi.sap.km._OdiSapAbapExecuteStoppable`
4. Click **Add Open Tool**.
5. The Open Tool appears as shown below.



6. Click OK.

6.3 Defining the Topology

You must define the two data servers used for SAP integration. The SAP ABAP Data Server and the FTP data server.

1. [Create the File Data Server](#)
2. [Create the SAP ABAP Data Server](#)

6.3.1 Create the File Data Server

This data server corresponds to the FTP server or File Server into which the extraction file will be pushed from SAP and picked up for SQL*Loader / JDBC driver.

6.3.1.1 Create a File Data Server

Create a File data server as described in "Creating a File Data Server" of *Connectivity and Knowledge Modules Guide for Oracle Data Integrator*. This section describes the parameters specific to SAP BW.

Depending on the chosen data transfer mode, this data server must point either to:

- An existing FTP server into which the extraction file will be pushed from SAP and picked up for loading or
- The shared folder into which the SAP system will write the extraction file and from which SQL*Loader/ ODI Flat File Driver will pick it up. This schema represents the folder in the FTP host where the extraction file will be pushed.

Note that the parameters for the data server depend on the data transfer mode.

- When transferring data through FTP, set the parameters as follows:
 - Host (Data Server): FTP server IP host name or IP address
 - User: Username to log into FTP server
 - Password: Password for the user

- When transferring data through a Shared Directory, set the parameters as follows:
 - Host (Data Server): n/a
 - User: n/a
 - Password: n/a
- For use with "LKM SAP BW to SQL" these additional parameters must be configured:
 - JDBC driver class: com.sunopsis.jdbc.driver.file.FileDriver
 - JDBC URL: jdbc:snps:dbfile?ENCODING=UTF8

The above URL is for SAP UNICODE systems. For non-UNICODE systems, please see details on ENCODING parameter in "Creating a File Data Server" of *Connectivity and Knowledge Modules Guide for Oracle Data Integrator*. The encoding chosen on this URL must match the code page used by the SAP Application Server.

See [Section 6.7.1, "File Transfer Configurations"](#) for more information.

6.3.1.2 Create the File Schema

In this File data server create a Physical Schema as described in "Creating a File Physical Schema" of the *Connectivity and Knowledge Modules Guide for Oracle Data Integrator*.

This schema represents the folder that is located either in the FTP host or the File Server. Note that this is the folder into which the extraction file will be pushed. Depending on the data transfer mode, specify the Data and Work Schemas as follows:

- For the FTP file transfer:
 - *Directory (Schema)*: Path on the FTP server to upload or download extraction files from the remote location. This path is used when uploading extraction files from the SAP BW system into the FTP server. It is also used by a remote agent to download the extraction files. Note that this path must use slashes and must end with a slash character.
 - *Directory (Work Schema)*: Local path on the FTP server's machine. This path is used by an agent installed on this machine to access the extraction files without passing via FTP. This access method is used if the FTP_TRANSFER_METHOD parameter of the LKM SAP BW to Oracle (SQLDR)/ LKM SAP BW to SQL is set to NONE. The Work Schema is a local directory location. As a consequence, slashes or backslashes should be used according to the operating system. This path must end with a slash or backslash.

Path names given on Data and Work schemas are not necessarily the same: the FTP server may provide access to a FTP directory named `/sapfiles` - the value for Directory (Schema) - while the files are accessed locally in `c:\inetpub\ftproot\sapfiles` - the value for Directory (Work Schema).
- For the Shared Directory transfer:
 - *Directory (Schema)*: Path (UNC) of the shared folder to write and read extraction files. SAP System writes the extraction files into this folder. It is also used by a remote agent to copy the extraction files to the ODI agent machine. Note that this path must use slashes or backslashes according to the operating system of the SAP Application Server and must end with a slash or backslash character.

- *Directory (Work Schema)*: Local path on the server's machine hosting the shared folder. This path is used by an agent installed on this machine to access the extraction files without passing through the shared folder. This access method is used if the FTP_TRANSFER_METHOD parameter of the LKM SAP BW to Oracle (SQLLDR)/ LKM SAP BW to SQL is set to FSMOUNT_DIRECT. The Work Schema is a local directory location. As a consequence, slashes or backslashes should be used according to the operating system. This path must end with a slash or backslash.

See [Section 6.7.1, "File Transfer Configurations"](#) for more information.

Create a File Logical Schema called `File Server` for SAP ABAP, and map it to the Physical Schema. The name of this Logical Schema name is predefined and must be `File Server` for SAP ABAP.

6.3.2 Create the SAP ABAP Data Server

This SAP ABAP data server corresponds to the SAP server from which data will be extracted.

6.3.2.1 Create the SAP ABAP Data Server

To configure a SAP ABAP data server:

1. Create a data server for the SAP ABAP technology using the standard procedure, as described in "Creating a Data Server" of the *Developing Integration Projects with Oracle Data Integrator*. This data server uses the SAP connection information.
2. Set the connection parameters:
 - Name: The name of the data server as it will appear in ODI. For example, `SAP_BW`.
 - User: SAP BW User, as provided by the SAP Administrator.
 - Password: This user's SAP BW Password. This password is case-sensitive.
3. Set Logon type specific connection parameters:
 - either for SAP Group Logons (on FlexField tab):
 - FlexField SAP System ID: The three-character, unique identifier of a SAP system in a landscape.
 - FlexField SAP Message Server Host: IP address/ host name of SAP Message Server.
 - FlexField SAP Message Server Port: port number or service name of SAP Message Server.
 - FlexField SAP Group: Name of SAP Logon Group.
 - or for SAP Server Logons:
 - Host (Data Server): SAP BW System IP Address or host name.
 - FlexField SAP Client Number: The three-digit number assigned to the self-contained unit which is called Client in SAP. A Client can be a training, development, testing or production client or represent different divisions in a large company.
 - FlexField SAP System Number: The two-digit number assigned to a SAP instance which is also called Web Application Server or WAS.

- FlexField SAP System ID: The three-character, unique identifier of a SAP system in a landscape.
4. Set SAP Secure Network Settings (optional):
 - FlexField "for SNC: Level of Security": 0 for "No SNC", 1 for "Authentication", 2 for "Integrity", 3 for "Encryption", 9 for "Maximum Security Settings Available"
 - FlexField "for SNC: Client Name": SNC Client Name
 - FlexField "for SNC: Partner Name": SNC Partner Name
 5. Set the flexfields values for this data server in the Flexfields tab.
 - SAP Language: Code of the language used when logging in. For example EN for English, DE for German.
 - SAP Router String: Router String. This parameter is optional and can be left empty.
 - SAP Character Set: The character set is only required if your SAP system is not a UNICODE system. For a complete list of character sets, see "Locale Data" in the *Oracle Database Globalization Support Guide*. For example, EE8ISO8859P2 for Croatian Data. For UNICODE systems, use UTF8.
 - SAP Temporary Directory: In case of FTP based file transfer (compared to shared folder transfer) data is extracted to the default temp directory of the SAP server. If needed, a different folder can be used instead.
 - SAP Extract Timeout: Maximum number of seconds ODI will wait for a SAP background job before it times out.
 - SAP BW Version: Enter the SAP BW Version as follows:
 - For SAP BW 7.0 and higher systems enter 700
 - For SAP BI 3.5 systems enter 350
 - SAP ERP Version & SAP ABAP Version: not used in SAP BW Connector.
 - SAP Allow ABAP Upload: set to 1 if ABAP code can be uploaded on this SAP System. Typically set to 0 for any non-development system. See [Section 6.7.3, "Controlling ABAP Uploading / ABAP code in production"](#) for more details.
 - SAP Allow ABAP Execute: set to 1, if ABAP code can be executed on this SAP System. See [Section 6.7.3, "Controlling ABAP Uploading / ABAP code in production"](#) for more details.
-
- Note:** The Test button for validating the SAP Connection and the FTP Connection definition is not supported.
-
- SAP Transport Layer Name: This transport layer is used by ODI to create any new transport requests. The default transport layer name is SAP. If the SAP system uses a different transport layer, this FlexField must be updated accordingly. Otherwise, any TR creation will fail.

Except for Data Server Name, all the parameters that you provide while defining the SAP Data Server should be provided by the SAP Administrators. See [Gathering SAP Connection Information](#) for more information about these parameters.

6.3.2.2 Create the SAP ABAP Schema

To configure a SAP ABAP schema:

1. Create a Physical Schema under the SAP ABAP data server as described in "Creating a Physical Schema" in *Administering Oracle Data Integrator*. This schema does not require any specific configuration. Only one physical schema is required under a SAP ABAP data server.
2. Create a Logical Schema for this Physical Schema as described in "Creating a Logical Schema" in *Administering Oracle Data Integrator* in the appropriate context.

6.4 Setting up the Project

Setting up a project using SAP BW features follows the standard procedure. See "Creating an Integration Project" of the *Developing Integration Projects with Oracle Data Integrator*.

Import the following KMs into your Oracle Data Integrator project:

- RKM SAP BW
- LKM SAP BW to Oracle (SQLDR)
- LKM SAP BW to SQL

In addition to these specific SAP BW KMs, import the standard Oracle LKMs, IKMs, and CKMs to perform data extraction and data quality checks with an Oracle database. See "Oracle Database" in the *Connectivity and Knowledge Modules Guide for Oracle Data Integrator* for a list of available KMs.

6.5 Creating and Reverse-Engineering a Model

This section contains the following topics:

- [Creating a SAP BW Model](#)
- [Reverse-Engineering a SAP BW Model](#)

6.5.1 Creating a SAP BW Model

Create an SAP BW Model based on the SAP ABAP technology and on the SAP ABAP logical schema using the standard procedure, as described in "Creating a Model" of the *Developing Integration Projects with Oracle Data Integrator*.

6.5.2 Reverse-Engineering a SAP BW Model

To perform a Customized Reverse-Engineering with the RKM SAP BW, use the usual procedure, as described in "Reverse-engineering a Model" of the *Developing Integration Projects with Oracle Data Integrator*. This section details only the fields specific to SAP BW:

1. In the Reverse Engineer tab of the SAP BW Model, select the RKM SAP BW.
2. For the RKM SAP BW, set the USE_GUI KM option to true.
3. Save the model.
4. Click **Reverse-Engineer** in the Model Editor toolbar.
5. The Tree Metadata Browser appears after the session is started. Select the data store object(s) to reverse.

6. Click **Reverse-Engineer** in the Tree Metadata Browser window.

The reverse-engineering process returns the selected data store objects as datastores.

Note: If the reverse-engineering is executed on a run-time agent, the USE_GUI option should be set to false. This option should be used only when the customized reverse-engineering is started using the agent built-in the Studio.

6.6 Designing a Mapping

To create a mapping loading SAP BW data into an Oracle staging area:

1. Create a mapping with source datastores from the SAP BW Model. This mapping should have an Oracle target or use an Oracle schema as the Staging Area.
2. Create joins, filters, and map attributes for your mapping.
3. In the Physical diagram of the mapping, select the access point for the SAP BW source data object(s). The Property Inspector opens for this object.
4. In the Loading Knowledge Module tab, select the LKM SAP BW to Oracle (SQLLDR).

6.7 Considerations for SAP BW Integration

This section includes the following topics:

- [File Transfer Configurations](#)
- [Execution Modes](#)
- [Controlling ABAP Uploading / ABAP code in production](#)
- [Managing ODI SAP Transport Requests](#)
- [SAP Packages and SAP Function Groups](#)
- [Log Files](#)
- [Limitation of the SAP BW Adapter](#)

6.7.1 File Transfer Configurations

The ODI SAP adapter extracts data using ABAP programs. For transferring the data from SAP system to the ODI agent the adapter supports two transfer modes and different configurations:

- [Transfer using a Shared Directory \(recommended\)](#)
- [FTP based Transfer](#)

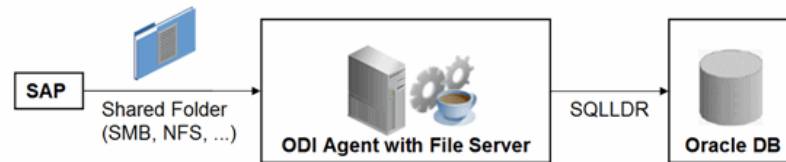
6.7.1.1 Transfer using a Shared Directory (recommended)

During the extraction process the ABAP programs write chunks of data into the data file in the shared folder. For better performance this shared folder should be located on the ODI agent machine. In this setup, for LKM SAP BW to Oracle (SQLLDR): SQL*Loader reads locally the data file and loads the data into the Oracle staging area. For LKM SAP BW to SQL: The ODI File Driver reads locally the data file and inserts the data using JDBC into a non-Oracle staging area.

If the folder is not located on the ODI agent machine, then the ODI agent first needs to copy the file from the shared folder to the agent for loading the data using SQL*Loader/ JDBC-Connection in the next step.

Configuration 1: Shared Folder is physically located on the ODI Agent machine (recommended)

Figure 6–1 Configuration 1



This configuration is used, when `FTP_TRANSFER_METHOD = FSMOUNT_DIRECT`. In this configuration the following data movements are performed:

1. The ABAP program extracts chunks of `FETCH_BATCH SIZE` records and writes them into a file in the shared folder.
2. for LKM SAP BW to Oracle (SQL*Loader): SQL*Loader reads the data file from this `TEMP_DIR` and loads the data into the Oracle staging area. For LKM SAP BW to SQL: The ODI File Driver reads the data file from this `TEMP_DIR` and inserts the data using JDBC into a non-Oracle staging area.

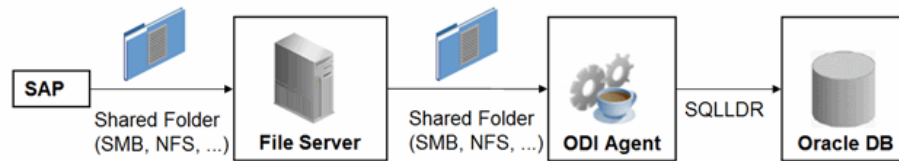
This configuration requires the following Topology settings:

1. Create a File data server pointing to the File server/ODI Agent machine:
 - Host (Data Server): n/a
 - User: n/a
 - Password: n/a
2. Under this File data server create a physical schema representing the shared folder in the File host server. Specify the Data and Work Schemas as follows:
 - *Data Schema*: Path (UNC) of the shared folder used by the ABAP program to write extraction files.
 - *Directory (Work Schema)*: Local path on the server's machine hosting the shared folder. This path is used by an agent and SQL*Loader installed on this machine to access the extraction files without passing via the shared folder.

Note: Temporary files such as `ctl`, `bad`, `dsc` will be created in a local temporary folder on the run-time agent. The default temporary directory is the system's temporary directory. On UNIX this is typically `/tmp` and on Windows `c:\Documents and Settings\\Local Settings\Temp`. This directory can be changed using the KM option `TEMP_DIR`.

Configuration 2: Shared folder is not physically located on the ODI Agent machine

Figure 6–2 Configuration 2



This configuration is used, when `FTP_TRANSFER_METHOD = FSMOUNT`. In this configuration the following data movements are performed:

1. The ABAP program extracts chunks of `FETCH_BATCH SIZE` records and writes them into a file in the shared folder.
2. The run-time agent copies the file into the directory given by `TEMP_DIR` option of the LKM.
3. for LKM SAP BW to Oracle (SQLLDR): `SQL*Loader` reads the data file from this `TEMP_DIR` and loads the data into the Oracle staging area. For LKM SAP BW to SQL: The ODI File Driver reads the data file from this `TEMP_DIR` and inserts the data using JDBC into a non-Oracle staging area.

This configuration requires the following Topology settings:

1. Create a File data server pointing to the file server into which the extraction file will be pushed from SAP and picked up from for `SQL*Loader`.

Set the parameters for this data server as follows:

- Host (Data Server): n/a
 - User: n/a
 - Password: n/a
2. In this File data server create a physical schema representing the shared folder. Specify the Data and Work Schemas as follows:
 - *Directory (Schema)*: Path (UNC) of the shared folder used by the ABAP program to write extraction files, and by the agent to copy the file.
 - *Directory (Work Schema)*: <undefined>. Leave this path blank, as data files are never accessed directly from the File server's file system.

Please note that data files will be copied to the run-time agent from the shared folder in a local temporary folder. The default temporary directory is the system's temporary directory. On UNIX this is typically `/tmp` and on Windows `c:\Documents and Settings\\Local Settings\Temp`. This directory can be changed using the KM option `TEMP_DIR`.

6.7.1.2 FTP based Transfer

At the end of the extraction process these ABAP programs will upload the data file to a FTP server. For better performance this FTP server should be located on the same machine as the run-time agent.

If the agent is not located on the same machine as the FTP server, it will download the file from the FTP server before loading it to the staging area `SQL*Loader/JDBC-Connection`. This download operation is performed using FTP, SFTP or SCP.

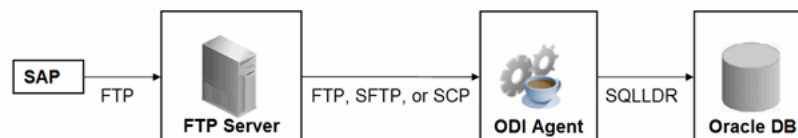
Figure 6–3 Configuration 1: FTP Server is installed on an ODI Agent machine


The configuration shown in [Figure 6–3](#) is used, when `FTP_TRANSFER_METHOD = NONE`. In this configuration the following data movements are performed:

1. The ABAP program extracts the data and uploads the data file to the FTP server.
2. For LKM SAP BW to Oracle (SQLLDR): SQL*Loader reads locally the data file and loads the data into the Oracle staging area. For LKM SAP BW to SQL: The ODI File Driver reads locally the data file and inserts the data using JDBC into a non-Oracle staging area.

This configuration requires the following Topology settings:

1. Create a File data server pointing to the FTP server:
 - Host (Data Server): FTP server host name or IP address.
 - User: Username to log into FTP server.
 - Password: Password for the user.
2. In this File data server create a physical schema representing the folder in the FTP host where the extraction file will be pushed. Specify the Data and Work Schemas as follows:
 - *Directory (Schema)*: Path on the FTP server for uploading SAP extraction files.
 - *Directory (Work Schema)*: Local path on the FTP server's machine containing the SAP extraction file. The agent and SQL*Loader/ODI Flat File Driver read the extraction files from this location.

Figure 6–4 Configuration 2: FTP Server is not installed on ODI Agent machine


The configuration shown in [Figure 6–4](#) is used, when `FTP_TRANSFER_METHOD` is `FTP`, `SFTP` or `SCP`. In this configuration the following data movements are performed:

1. The ABAP program extracts the data and uploads the data file to the FTP server.
2. The ODI agent downloads the file from the FTP server into the directory given by KM Option `TEMP_DIR`.
3. For LKM SAP BW to Oracle (SQLLDR): SQL*Loader reads the data file from this `TEMP_DIR` and loads the data into the Oracle staging area. For LKM SAP BW to SQL: The ODI File Driver reads the data file from this `TEMP_DIR` and inserts the data using JDBC into a non-Oracle staging area.

This configuration requires the following Topology settings:

1. Create a File data server pointing to the FTP server:
 - Host (Data Server): FTP server host name or IP address.
 - User: User name to log into FTP server.
 - Password: Password for the user.
2. In this File data server create a physical schema representing the folder in the FTP host where the extraction file will be pushed. Specify the Data and Work Schemas as follows:
 - *Directory (Schema)*: Path on the FTP server for uploading SAP extraction files.
 - *Directory (Work Schema)*: <undefined>; this path is left blank, as data files are never accessed directly from the FTP server's file system.

Considerations and Limitations:

The FTP based data transfer uses the widely spread (S)FTP file transfer and requires all data to be held in SAP's application server memory before transfer. Therefore the required memory per SAP session increases with the amount of data extracted and will set an upper limit to the data volume. This upper limit can be adjusted to a certain extend by increasing the sessions memory settings in SAP.

The required setup for the shared folder based configuration is slightly more complex, but it removes the need for all data to fit into SAP AS' memory and is therefore the recommended extraction method.

6.7.2 Execution Modes

Background Processing

By default the generated ABAP code will be deployed as an ABAP report. At execution this report is submitted to the SAP scheduler for background processing. The job status is then monitored by ODI.

The KM option `JOB_CLASS` defines the priority of the background Job. Valid values (corresponding to SAP `JOB_CLASS` settings) are:

`JOB_CLASS = A` is Highest Priority

`JOB_CLASS = B` is Normal Priority

`JOB_CLASS = C` is Lowest Priority

Dialog Mode Processing

For backwards compatibility the KM option `BACKGROUND_PROCESSING` can be set to false. The generated ABAP code will then be deployed as an RFC. At execution this RFC is called by ODI to extract the data.

Dialog mode processing has been deprecated and is currently supported for backwards compatibility only. In future releases dialog processing may be removed entirely.

6.7.3 Controlling ABAP Uploading / ABAP code in production

During development ODI generates ABAP code and uploads it into the SAP development system with every mapping execution. This automatic code upload allows quick development cycles.

Once a Mapping or Package has been unit tested and is ready to be migrated out of the development environment, the generated SAP ABAP code has to be transported to the respective SAP system using SAP's CTS (Change and Transport System) like any other SAP ABAP code. This is standard SAP practice. To facilitate this task, SAP transport requests are automatically created during upload into development. Please contact your SAP administrator for transporting generated ODI SAP ABAP programs.

In case you are working with distinct ODI repositories for dev, test and production, please make sure that your ODI scenario matches the ODI ABAP code of the respective SAP system. That is, you have to transport the SAP ABAP code using SAP CTS from your SAP development system to your SAP QA system *and* transport the ODI scenario (which has generated the transported ABAP code) from your ODI development repository to your ODI QA repository. Please see Chapter 13, "Working with Scenarios" of *Developing Integration Projects with Oracle Data Integrator* for details on how to transport ODI scenarios.

Once outside of development ODI should no longer upload ABAP code, as the ABAP code has been transported by SAP's CTS and such non-development systems usually do not allow ABAP uploading.

Even though uploading can be explicitly turned off by setting the LKM option `UPLOAD_ABAP_CODE` to No, it usually is turned off using the FlexField "SAP Allow ABAP Upload" defined on the SAP data server in ODI Topology: The ABAP code is only uploaded, if both the LKM option `UPLOAD_ABAP_CODE` and the Flexfield `SAP Allow ABAP Upload` are set to Yes. For disabling any upload into production systems it is sufficient to set the Flexfield "SAP Allow ABAP Upload" to 0 in Topology.

Tip: To configure a mapping that uploads the ABAP code in development but skips the upload in QA or production:

1. Set the KM option `UPLOAD_ABAP_CODE` set to Yes in all mappings
2. Configure the SAP data servers in the Topology as follows:
 - Set the Flexfield *SAP Allow ABAP Upload* to 1 for all SAP development systems
 - Set the Flexfield *SAP Allow ABAP Upload* to 0 for all other SAP systems

Note: Before starting the extraction process, ODI verifies that the mapping/scenario matches the code installed in SAP. If there is a discrepancy - for example, if the scenario was modified but the ABAP code was not re-uploaded - an exception is thrown.

In some situations it may be desirable just to install the Mapping's ABAP extraction code and not to extract any data, such as for an automated installation. In this case all mappings can be linked inside a package with the KM option `EXECUTE_ABAP_CODE` set to `False` in every mapping. Executing this package will then install all ABAP code, but will not perform any execution.

To avoid the modification of all mappings (setting `EXECUTE_ABAP_CODE` to `False` as described above), you can instead disable all SAP ABAP executions by using the FlexField **SAP Allow ABAP Execute** on the ODI DataServer. If this FlexField is *disabled*, the ABAP code is not executed even if the KM option `EXECUTE_ABAP_CODE` is set to `True`.

Manual Upload

In some cases, automatic upload may not be allowed even in development systems. For such situations the KM option `MANUAL_ABAP_CODE_UPLOAD` allows manual uploads. If set to `true`, ODI will create a text file containing the generated ABAP code. By default, the name of the text file is similar to `REPORT_ZODI_<Mapping Id>_<SourceSet Id>.ABAP`. This code is handed over to the SAP administrator, who will install it. Once installed the ODI mapping can be executed with `MANUAL_ABAP_CODE_UPLOAD` and `UPLOAD_ABAP_CODE` both set back to `false`.

6.7.4 Managing ODI SAP Transport Requests

During development, ABAP code is uploaded to the SAP system with every mapping execution. More precisely:

An ODI mapping extracting SAP data generates one or several ABAP extraction programs (e.g. when join location is set to staging and consequently two extraction jobs are created). By default all ABAP extraction programs of one mapping are assigned to one SAP function group. The ABAP extraction programs for a different mapping will be assigned to a different SAP function group. The default function group name is similar to `ZODI_FGR_<Mapping Id>`.

During upload a SAP CTS transport request is created for each ODI Mapping (for each SAP function group). This allows granular deployment of the generated ODI ABAP extraction programs via SAP CTS.

Grouping (Background Processing)

When the ABAP code of multiple ODI Mappings should be grouped into a single transport request for more coarse-grained deployment control, the following steps are needed:

- Set KM option `SAP_REPORT_NAME_PREFIX` for a common prefix for all ABAP reports. For example, `ZODI_DWH_SALES_` on all mappings.
- Choose a KM option `ABAP_PROGRAM_NAME` for every individual mapping. For example, `LOAD01`, `LOAD02`, etc.

These sample settings would result in a single transport request containing the ABAP reports called `ZODI_DWH_SALES_LOAD01`, `ZODI_DWH_SALES_LOAD02`, etc.

Grouping (Dialog mode processing, deprecated)

When the ABAP code of multiple ODI Mappings should be grouped into a single transport request for more coarse-grained deployment control, the KM option `SAP_FUNCTION_GROUP` for all LKMs in these mappings can be set to a user defined value, e.g. `ZODI_FGR_DWH_SALES`. This then leads to ODI generating all ABAP extraction programs into the same SAP function group which is then attached to a single transport request. For valid function group names at your site please contact your SAP administrator.

Tip: The name of the generated ABAP extraction programs is by default similar to `ZODI_<Mapping Id>_<SourceSet Id>`. This ensures convenient development due to unique program names. While the `MappingId` never changes, certain changes to an ODI mapping can cause the `SourceSetId` to change and consequently cause the respective extraction program name to change. Therefore it is recommended to use user-defined program names, once development stabilizes. ABAP program names can be set by defining a value for LKM option `ABAP_PROGRAM_NAME`, e.g. `ZODI_DWH_SALES_DATA01`. Please contact your SAP administrator for applicable naming conventions.

Transport Request Description

When ODI creates a transport request, it will set the transport request description to the text provided in KM option `SAP_TRANSPORT_REQUEST_DESC` and applies to function group defined in KM option `FUNCTION_GROUP`.

By default the ODI Step name (which is usually the mapping name) will be used.

Code generation expressions like `ODI:<%=odiRef.getPackage("PACKAGE_NAME")%>` may be useful when grouping several mappings into one SAP function group/ SAP transport request.

6.7.5 SAP Packages and SAP Function Groups

SAP Packages

All SAP objects installed by the ODI SAP Adapter are grouped into SAP packages :

- `ZODI_RKM_PCKG` contains any RKM related objects. These objects are used during development phase.
- `ZODI_LKM_PCKG` contains any extraction programs generated by any SAP LKM.

If requested by the SAP administrator, these default names can be overwritten using the KM options `SAP_PACKAGE_NAME_ODI_DEV` and `SAP_PACKAGE_NAME_ODI_PROD`. These values must be set during the first-time installation. Later changes require a reinstallation of the ODI SAP Adapter with a prior uninstallation.

Please note that LKM option `SAP_PACKAGE_NAME` must always use the same value given during first-time RKM installation. This means that when non-default values are used during first-time RKM installation, **all** mappings must set the LKM option `SAP_PACKAGE_NAME` to the non-default value.

SAP Function Groups

All SAP function modules are grouped into SAP function groups:

- `ZODI_FGR` contains any function modules needed during development, e.g. for retrieving metadata. These function modules are installed by the RKM during first-time installation.

The default name can be overwritten during first-time RKM installation using RKM option `SAP_FUNCTION_GROUP_ODI_DEV`.

- `ZODI_FGR_PROD` contains any function modules needed at runtime in production, e.g. for monitoring ABAP report execution status. These function modules are installed by the RKM during first-time installation.

The default name can be overwritten during first-time RKM installation using RKM option `SAP_FUNCTION_GROUP_ODI_PROD`.

- `ZODI_FGR_PROD_...` contains any data extraction function modules. Such function modules are generated by the LKM when using deprecated dialog mode (`BACKGROUND_PROCESSING = false`).

By default every mapping uses its own function group. The default values can be overwritten using the LKM option `SAP_FUNCTION_GROUP`, which is independent of the two function group names mentioned above.

See [Section 6.7.4, "Managing ODI SAP Transport Requests"](#) for more information.

6.7.6 Log Files

During RKM and LKM execution many messages are logged in ODI's logging facility. These messages in the ODI log files and other log files may contain valuable details for troubleshooting. See section "Runtime Logging for ODI components" of the *Administering Oracle Data Integrator* for details about the ODI's logging facility. [Table 6–2](#) describes the different log files and their usage:

Table 6–2 Log Files

Default Log File Name	KM / Phase	Content
<ODI Logs>	RKM	Execution Log of metadata retrieval
<ODI Logs>	RKM	Information about first time installation of SAP RFC for RKM
<ODI Logs>	LKM - Generation Time	Information about code generation for ABAP extractor
<ODI Logs>	LKM - Runtime	Information about installation of ABAP extractor
<ODI Logs>	LKM - Runtime	Information about Delta Extraction
<System Temp Dir or local FTP dir>/ ZODI_<Mapping Id>_<SrcSet>_<Context>.log	LKM - Runtime	SQL*Loader log file
<System Temp Dir or local FTP dir>/ ZODI_<Mapping Id>_<SrcSet>_<Context>.out	LKM - Runtime	OS std output during SQL*Loader execution, may contain information, e.g. when SQL*Loader is not installed
<System Temp Dir or local FTP dir>/ ZODI_<Mapping Id>_<SrcSet>_<Context>.err	LKM - Runtime	OS error output during SQL*Loader execution, may contain information, e.g. when SQL*Loader is not installed

6.7.7 Limitation of the SAP BW Adapter

The SAP ABAP BW adapter has the following limitations:

- The **Test** button for validating SAP Connection definition in ODI's Topology manager is not supported.
- The SAP BW data store type (InfoCube, InfoObject, ODS/DSO, OpenHub, Hierarchy, and Text Table) cannot be changed after a table has been reverse-engineered.
- The SAP ABAP KMs only support Ordered Joins.
- Full Outer join and Right outer joins are not supported.
- In one-to-many relationships (InfoCube and associated InfoObject join), the first data target should be InfoCube and then InfoObjects and its TextTables.
- InfoCubes can be left-outer joined to multiple IOs on source.

Note: ■ Only a single one of these InfoObjects can be joined to its Text table on source.

- Joining multiple InfoObjects to their respective Text tables is possible on staging.
-

- Hierarchy datastores cannot be joined on source with any other SAP BW objects.
- Text datastores of InfoObjects having no master data cannot be joined on source with any other SAP BW objects.
- OpenHub datastores cannot be joined on source with any other SAP BW objects.
- Only attribute RSHIENM can be filtered on using a constant string value, for example `HIER_OGL_ACCOUNT.RSHIENM = 'MYHIER1'`

This chapter describes how to work with SAP ERP Knowledge Modules in Oracle Data Integrator.

This chapter includes the following sections:

- [Section 7.1, "Introduction"](#)
- [Section 7.2, "Installation and Configuration"](#)
- [Section 7.3, "Defining the Topology"](#)
- [Section 7.4, "Setting up the Project"](#)
- [Section 7.5, "Creating and Reverse-Engineering a Model"](#)
- [Section 7.6, "Designing a Mapping"](#)
- [Section 7.7, "Considerations for SAP ERP Integration"](#)

7.1 Introduction

The SAP ERP Knowledge Modules let Oracle Data Integrator connect to SAP-ERP system using SAP Java Connector (SAP JCo) libraries. These adapters allows mass data extraction from SAP-ERP systems.

If this is the first time you are using the SAP ERP adapter, it is recommended to review *Getting Started with SAP ABAP ERP Adapter for Oracle Data Integrator*.

It contains the complete pre-requisites list as well as step-by-step instructions including SAP connection testing.

7.1.1 Concepts

The SAP ERP Knowledge Modules for Oracle Data Integrator use mature integration methods for SAP-ERP system, in order to:

- Reverse-Engineer SAP ERP metadata
- Extract and load data from SAP ERP system (source) to an Oracle or non-Oracle Staging Area

7.1.2 Knowledge Modules

Oracle Data Integrator provides the Knowledge Modules listed in [Table 7-1](#) for handling SAP ERP data.

The Oracle Data Integrator SAP ERP Knowledge Modules provide integration from SAP ERP systems using SAP JCo libraries. This set of KMs has the following features:

- Reads SAP data from SAP ERP system
- Loads this SAP data into Oracle or non-Oracle Staging Area
- Reverse-engineers SAP Metadata and proposes a tree browser to select only the required Metadata
- Uses flexfields to map the SAP table types (Transparent, Cluster, and Pool)

Table 7–1 SAP ERP Knowledge Modules

Knowledge Module	Description
LKM SAP ERP to Oracle (SQLLDR)	Extracts data from SAP ERP into a flat file and then loads it into Oracle Staging Area using the SQL*LOADER command line utility.
LKM SAP ERP to SQL	Extracts data from SAP ERP into a flat file and then loads it into a Staging Area using a JDBC connection.
RKM SAP ERP	Reverse-engineering Knowledge Module to retrieve SAP specific metadata for modules, application components, tables, columns, primary keys, foreign keys and indexes.
RKM SAP ERP Connection Test	This RKM is used for testing the SAP connection from Oracle Data Integrator. See Appendix B.2, "SAP Connection Test" for more information.

7.1.3 Overview of the SAP ABAP Integration Process

The RKM SAP ERP enables Oracle Data Integrator to connect to SAP ERP system using SAP JCo libraries and perform a customized reverse-engineering of SAP metadata.

The LKM SAP ERP to Oracle (SQLLDR) and LKM SAP ERP to SQL are in charge of extracting and loading data from SAP ERP system (Source) to an Oracle or non-Oracle Staging Area.

Note: Access to SAP ERP is made through ABAP. As a consequence, the technology used for connecting is SAP ABAP.

7.1.3.1 Reverse-Engineering Process

Reverse-engineering uses the RKM SAP ERP.

This KM automatically installs dedicated RFC programs to retrieve SAP metadata. It extracts the list of all SAP tables and optionally displays this list in a Metadata Browser graphical interface. The user selects from this list the tables to reverse-engineer.

The reverse-engineering process retrieves tables, primary keys, foreign keys and indexes.

7.1.3.2 Integration Process

Data integration from SAP is handled by the LKM SAP ERP to Oracle (SQLLDR) and the LKM SAP ERP to SQL.

The LKM SAP ERP to Oracle (SQLLDR) is used for mappings sourcing from SAP via ABAP and having a Staging Area located in an Oracle Database and the LKM SAP ERP to SQL is used for non-Oracle staging areas.

The KM first generates optimized ABAP code corresponding to the extraction process required for a given mapping. This code includes filters and joins that can be

processed directly in the source SAP server. This ABAP program is automatically uploaded and is executed using the OdiSAPAbapExecute tool to generate an extraction file in SAP.

The KM then transfers this extraction file either to a pre-configured FTP server or to a shared directory. This file is then either downloaded from this server using FTP, SFTP, SCP or copied to the machine where the ODI Agent is located, and is finally loaded either using SQL*Loader or using a JDBC connection into the staging area. The agent can also directly read the extraction file on the FTP server's disk. See [Section 7.7.1, "File Transfer Configurations"](#) for more information.

The rest of the integration process (data integrity check and integration) is managed with other Oracle Data Integration KMs.

7.2 Installation and Configuration

Make sure you have read the information in this section before you start working with the SAP ERP data:

- [System Requirements and Certification](#)
- [Technology Specific Requirements](#)
- [Connectivity Requirements](#)

7.2.1 System Requirements and Certification

Before performing any installation you should read the system requirements and certification documentation to ensure that your environment meets the minimum installation requirements for the products you are installing.

The list of supported platforms and versions is available on Oracle Technical Network (OTN):

<http://www.oracle.com/technology/products/oracle-data-integrator/index.html>.

7.2.2 Technology Specific Requirements

Some of the Knowledge Modules for SAP ERP use specific features of SAP and of the Oracle database. This section lists the requirements related to these features.

- A JCo version compatible with adapter must be used. The list of supported JCo versions is available from the Oracle Technology Network (OTN). See [Section 7.2.1, "System Requirements and Certification"](#) for more information.
- A JVM version compatible with both Oracle Data Integrator and JCo must be used.
- The adapter supports two transfer modes for transferring data from SAP system to the ODI agent: data transfer using a Shared Directory and data transfer through FTP. For details and restrictions, see [Section 7.7.1, "File Transfer Configurations"](#).

Depending on the chosen file transfer mode the following requirements must be met:

- **Data transfer through a Shared Directory (recommended transfer method)**

The LKM SAP ERP to Oracle (SQL*Loader) requires a folder that is shared between the SAP system and the ODI agent. The SAP application server transfers the data by writing it out into a folder that is accessible from the SAP system and the ODI agent machine. This is typically done by sharing a folder

of the ODI agent machine with the SAP system. Note that the shared folder does not necessarily have to be located on the ODI agent machine. A shared folder on a third machine is also possible, as long as the shared folder is accessible to both the ODI agent machine and the SAP system.

Note: For security reasons, folders located on the SAP server should not be shared. You should instead share a folder located on the ODI agent machine with the SAP system, or use a third machine as the shared file server.

The shared folder must be accessible to SAP system and not just to the underlying operating system. This means that the folder needs to be declared in SAP transaction AL11 and the folder opens successfully in AL11.

– **Data transfer through FTP**

LKM SAP ERP to Oracle (SQLLDR) requires a FTP server to upload data from the SAP ERP system. This data is either read locally by the agent executing the mapping (when this agent runs on the FTP server machine), or remotely (when this agent is located on a different machine than the FTP server). This FTP server must be accessible over the network from both the SAP ERP machine and the agent machine.

- For "LKM SAP ERP to Oracle (SQLLDR)" only: SQL*Loader is required on the machine running the agent when executing mappings using LKM SAP ERP to Oracle (SQLLDR). SQL*Loader is used for loading data extracted from SAP to the Oracle staging area.

7.2.3 Connectivity Requirements

Oracle Data Integrator connects to the SAP-ERP system hosting the SAP ERP data using JCo. It also uses a FTP Server or a Shared Directory to host the data extracted from the SAP system.

This section describes the required connection information:

- [Installing and Configuring JCo](#)
- [Installing ODI SAP Components into SAP System](#)
- [Validating the SAP Environment Setup](#)
- [Gathering SAP Connection Information](#)
- [Gathering FTP Connection Information](#)
- [Gathering Shared Directory Information](#)
- [Adding the Open Tool](#)

7.2.3.1 Installing and Configuring JCo

The SAP adapter uses JCo to connect to the SAP system. JCo must be configured before proceeding with the project.

To install and configure JCo:

1. Download a supported JCo version for your configuration from <http://service.sap.com/connectors>. Check the supported JCo version in the Compatibility Matrix available at Oracle Technology Network:

http://www.oracle.com/technology/products/oracle-data-integrator/10.1.3/htdocs/documentation/odi_certification.xls

Notes:

- A minimum version of JCo 3.0.2 is required
 - Choose the SAP JCo package matching your operating system and your system architecture (32/64Bit). E.g. if you are running ODI inside a 32-Bit JVM, you must download the 32-Bit SAP JCo, even if the CPU and OS are 64-Bit. Mixing 32-bit and 64-bit architecture is not possible due to native libraries required by SAP JCo and will result in connection failure.
 - `odi.conf` contains the JDK path used for ODI Studio.
-
-

2. Unzip the appropriate distribution package into a temporary directory `<sapjco-install-path>`.
3. Follow the installation instructions provided in the JCo documentation (`<sapjco-install-path>/javadoc/installation.html`) for your platform.
4. Copy the required files (`sapjco3.jar` and `sapjco3.dll/.so`) into the `<ODI_HOME>/odi/sdk/lib` directory (ODI Studio, ODI Standalone Agent) and into the `<WLS_DOMAIN>/lib` directory (JEE Agent).
5. Restart the ODI Components using SAP (ODI Studio, Standalone Agent)
6. Check the JCo installation.

Note: For SAP Secure Network Connections, please ensure that:

- the environment variable `SECUDIR` points to the directory containing any certificate files.
- the environment variable `SNC_LIB` points to the directory containing `sapcrypto.dll/libsapcrypto.so` for Linux.

Please contact your SAP Administrator for more details on certificates and Crypto Libraries.

7.2.3.2 Installing ODI SAP Components into SAP System

The ODI SAP adapter communicates with the SAP System using a few ODI SAP RFCs. These RFCs are installed by your SAP Basis team using SAP Transport requests. Please contact your SAP administrators for installing the ODI SAP Components and assigning the required SAP user authorizations following the instructions given in [Appendix C.2, "Installing ODI SAP Components"](#).

7.2.3.3 Validating the SAP Environment Setup

[Appendix C.3, "Validating the ODI SAP Setup"](#) contains instructions for some basic validation of the SAP environment for the use with the ODI SAP Adapter. Please ask your SAP Basis team to run all validations and provide back the validation results like screen shots and confirmations.

7.2.3.4 Gathering SAP Connection Information

In order to connect to the SAP ERP system, you must request the following information from your SAP administrators:

- either for SAP Group Logons:
 - SAP System ID: The three-character, unique identifier of a SAP system in a landscape.
 - Message Server: IP address/ host name of SAP Message Server.
 - Message Server Port: port number or service name of SAP Message Server.
 - Group/Server: Name of SAP Logon Group.
- or for SAP Server Logons:
 - SAP ERP System IP Address or Hostname: IP address/ host name of the host on which SAP is running.
 - SAP Client Number: The three-digit number assigned to the self-contained unit which is called Client in SAP. A Client can be a training, development, testing or production client or represent different divisions in a large company.
 - SAP System Number: The two-digit number assigned to a SAP instance which is also called Web Application Server or WAS.
 - SAP System ID: The three-character, unique identifier of a SAP system in a landscape.
- SAP ERP System IP Address or Hostname: IP address/ Hostname of the host on which SAP is running.
- SAP User: SAP User is the unique user name given to a user for logging on the SAP System.
- SAP Password: Case-sensitive password used by the user to log in.
- SAP Language: Code of the language used when logging in For example: EN for English, DE for German.
- SAP Client Number: The three-digit number assigned to the self-contained unit which is called Client in SAP. A Client can be a training, development, testing or production client or represent different divisions in a large company.
- SAP System Number: The two-digit number assigned to a SAP instance which is also called Web Application Server or WAS.
- SAP System ID: The three-character, unique identifier of a SAP system in a landscape.
- SAP SNC Connection Properties (optional) SAP Router String (optional): SAP is enhancing security through SNC and SAP router. It is used when these securities are implemented.
- SAP Transport Layer Name: This string uniquely identifies a transport layer in a SAP landscape. It allows ODI to create transport requests for later deployment in SAP.
- SAP Temporary Directory (`SAP_TMP_DIR`): This parameter is used to define the custom work directory on the SAP system. It applies only in case of FTP transfer mode. This path is used for temporary files during extraction. The default value (FlexField value is left empty) is to use the `DIR_HOME` path defined in the SAP profile.

Specify a directory on the SAP application server to be used for temporary files. The path must end on slash/ backslash depending on the OS the SAP application server runs on. The ODI SAP user must have read and write privileges on this directory.

Caution: The default value (empty FF value) must be avoided for any critical SAP system, as an excessive temp file during extraction could fill up the respective file system and can cause serious issues on the SAP system. Even for SAP development systems it is strongly recommended to override the default value.

- SAP ABAP Version: The version of the SAP system.
- SAP Character Set: The character set is only required if your SAP system is not a UNICODE system. For a complete list of character sets, see "Locale Data" in the *Oracle Database Globalization Support Guide*. For example, EE8ISO8859P2 for Croatian Data. For UNICODE systems, use UTF8.

Note: All the connection data listed above (except SAP SNC Connection Properties and SAP Router String) are mandatory and should be requested from the SAP Administrators. You may consider requesting support during connection setup from your SAP administrators.

7.2.3.5 Gathering FTP Connection Information

Gathering FTP connection information only applies if you plan to transfer data using FTP. The SAP ERP system will push data to a server using the FTP protocol. Collect the following information from your system administrator:

- FTP server name or IP address
- FTP login ID
- FTP login password
- Directory path for storing temporary data files

Validate that the FTP server is accessible both from SAP System and from ODI agent machine.

7.2.3.6 Gathering Shared Directory Information

Gathering Shared Directory information only applies, if you plan to transfer data through a shared directory. The SAP system will push data to a shared folder. For later setup, gather the following information from your system administrator:

- (UNC) path name of the shared folder

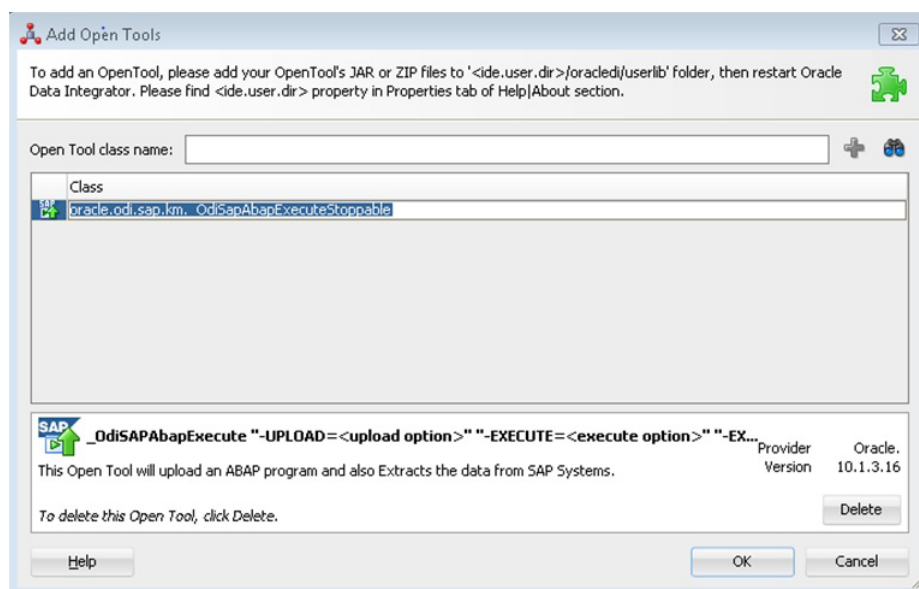
Validate that the shared folder is accessible from both the SAP System and the ODI agent machine and does not require any interactive authentication to be accessed.

Please note that the shared folder must be accessible from the SAP system using the `<sid>adm` user and from the operating system user that starts the ODI agent.

7.2.4 Adding the Open Tool

The complete process of installing and adding an Open Tool to ODI is described in *Oracle Data Integrator Tool Reference*. This section details only the SAP ABAP specific steps.

1. Connect to Designer.
2. Select **File > Add/Remove Open Tools...**
3. In the Add/remove Open Tools window, enter the following name in the Open Tool class name field:
`oracle.odi.sap.km._OdiSapAbapExecuteStoppable`
4. Click **Add Open Tool**.
5. The Open Tool appears as shown below.



6. Click **OK**.

7.3 Defining the Topology

You must define the two data servers used for SAP integration. The SAP ABAP Data Server and the File Data server.

1. [Create the File Data Server](#)
2. [Create the SAP ABAP Data Server](#)

7.3.1 Create the File Data Server

This data server corresponds to the FTP server or File Server into which the extraction file will be pushed from SAP and picked up for Loading.

7.3.1.1 Create a File Data Server

Create a File data server as described in "Creating a File Data Server" of the *Connectivity and Knowledge Modules Guide for Oracle Data Integrator*. This section describes the parameters specific to SAP ERP.

Depending on the chosen data transfer mode, this data server must point either to:

- An existing FTP server into which the extraction file will be pushed from SAP and picked up for Loading or
- The shared folder into which the SAP system will write the extraction file and from which SQL*Loader/ ODI Flat File Driver will pick it up. This schema represents the folder in the FTP host where the extraction file will be pushed.

Note that the parameters for the data server depend on the data transfer mode.

- When transferring data through FTP, set the parameters as follows:
 - Host (Data Server): FTP server IP host name or IP address
 - User: Username to log into FTP server
 - Password: Password for the user
- When transferring data through a Shared Directory, set the parameters as follows:
 - Host (Data Server): n/a
 - User: n/a
 - Password: n/a
- For use with "LKM SAP ERP to SQL" these additional parameters must be configured:
 - JDBC driver class: com.sunopsis.jdbc.driver.file.FileDriver
 - JDBC URL: jdbc:snps:dbfile?ENCODING=UTF8

The above URL is for SAP UNICODE systems. For non-UNICODE systems, please see details on ENCODING parameter in "Creating a File Data Server" of the *Connectivity and Knowledge Modules Guide for Oracle Data Integrator*. The encoding chosen on this URL must match the code page used by the SAP Application Server.

See [Section 7.7.1, "File Transfer Configurations"](#) for more information.

7.3.1.2 Create the File Schema

In this File data server create a Physical Schema as described in "Creating a File Physical Schema" of the *Connectivity and Knowledge Modules Guide for Oracle Data Integrator*.

This schema represents the folder that is located either in the FTP host or the File Server. Note that this is the folder into which the extraction file will be pushed. Depending on the data transfer mode, specify the Data and Work Schemas as follows:

- For the FTP file transfer:
 - *Directory (Schema)*: Path on the FTP server to upload or download extraction files from the remote location. This path is used when uploading extraction files from the SAP ERP system into the FTP server. It is also used by a remote agent to download the extraction files. Note that this path must use slashes and must end with a slash character.
 - *Directory (Work Schema)*: Local path on the FTP server's machine. This path is used by an agent installed on this machine to access the extraction files without passing via FTP. This access method is used if the FTP_TRANSFER_METHOD parameter of the LKM SAP ERP to Oracle (SQLLDR)/ LKM SAP ERP to SQL is set to NONE. The Work Schema is a local directory location. As a

consequence, slashes or backslashes should be used according to the operating system. This path must end with a slash or backslash.

Path names given on Data and Work schemas are not necessarily the same: the FTP server may provide access to a FTP directory named `/sapfiles` - the value for Directory (Schema) - while the files are accessed locally in `c:\inetpub\ftproot\sapfiles` - the value for Directory (Work Schema).

- For the Shared Directory transfer:
 - *Directory (Schema)*: Path (UNC) of the shared folder to write and read extraction files. SAP System writes the extraction files into this folder. It is also used by a remote agent to copy the extraction files to the ODI agent machine. Note that this path must use slashes or backslashes according to the operating system of the SAP Application Server and must end with a slash or backslash character.
 - *Directory (Work Schema)*: Local path on the server's machine hosting the shared folder. This path is used by an agent installed on this machine to access the extraction files without passing through the shared folder. This access method is used if the `FTP_TRANSFER_METHOD` parameter of the LKM SAP ERP to Oracle (SQLLDR)/ LKM SAP ERP to SQL is set to `FSMOUNT_DIRECT`. The Work Schema is a local directory location. As a consequence, slashes or backslashes should be used according to the operating system. This path must end with a slash or backslash.

See [Section 7.7.1, "File Transfer Configurations"](#) for more information.

Create a File Logical Schema called `File Server` for SAP ABAP, and map it to the Physical Schema. The name of this Logical Schema name is predefined and must be `File Server` for SAP ABAP.

7.3.2 Create the SAP ABAP Data Server

This SAP ABAP data server corresponds to the SAP server from which data will be extracted.

7.3.2.1 Create the SAP ABAP Data Server

To configure a SAP ABAP data server:

1. Create a data server for the SAP ABAP technology using the standard procedure, as described in "Creating a Data Server" of the *Developing Integration Projects with Oracle Data Integrator*. This data server uses the SAP connection information.
2. Set connection parameters:
 - Name: The name of the data server as it will appear in ODI. For example, `SAP_ERP`
 - User: SAP ERP User, as provided by the SAP Administrator
 - Password: This user's SAP ERP Password. This password is case-sensitive.
3. Set Logon type specific connection parameters:
 - either for SAP Group Logons (on FlexField tab)
 - FlexField SAP System ID: The three-character, unique identifier of a SAP system in a landscape.
 - FlexField SAP Message Server Host: IP address/ host name of SAP Message Server.

- FlexField SAP Message Server Port: port number or service name of SAP Message Server.
- FlexField SAP Group: Name of SAP Logon Group.
- or for SAP Server Logons
 - Host (Data Server): SAP ERP System IP Address or host name.
 - FlexField SAP Client Number: The three-digit number assigned to the self-contained unit which is called Client in SAP. A Client can be a training, development, testing or production client or represent different divisions in a large company.
 - FlexField SAP System Number: The two-digit number assigned to a SAP instance which is also called Web Application Server or WAS.
 - FlexField SAP System ID: The three-character, unique identifier of a SAP system in a landscape.
- 4. Set SAP Secure Network Settings (optional):
 - FlexField "for SNC: Level of Security": 0 for "No SNC", 1 for "Authentication", 2 for "Integrity", 3 for "Encryption", 9 for "Maximum Security Settings Available"
 - FlexField "for SNC: Client Name": SNC Client Name
 - FlexField "for SNC: Partner Name": SNC Partner Name
- 5. Set other flexfields values for this data server in the Flexfields tab.
 - SAP Language: Code of the language used when logging in. For example EN for English, DE for German.
 - SAP Router String: Router String. This parameter is optional and can be left empty.
 - SAP Character Set: Character set of the SAP system. This information is used for SQLLDR when loading data. For UNICODE SAP systems, use the default value UTF8. For a complete list of character sets, see "Locale Data" in the *Oracle Database Globalization Support Guide*. For example, EE8ISO8859P2 for Croatian Data.
 - For a complete list of character sets, please refer to Oracle Database Globalization Support Guide: Locale Data: Character Sets. For example, EE8ISO8859P2 for Croatian Data."
 - SAP Temporary Directory: In case of FTP based file transfer (compared to shared folder transfer) data is extracted to the default temp directory of the SAP server. If needed, a different folder can be used instead.
 - SAP Extract Timeout: Maximum number of seconds ODI will wait for a SAP background job before it times out.
 - SAP ABAP Version: Enter the SAP ABAP version as follows:
 - For SAP 4.6C enter 46C
 - For SAP 4.7 enter 620
 - For SAP ECC 5.0 enter 640
 - For SAP ECC 6.0 enter 700
 - SAP ERP Version: Reserved for future use.

- SAP Allow ABAP Upload: set to 1, if ABAP code can be uploaded on this SAP System. Typically set to 0 for any non-development system. See [Section 7.7.3, "Controlling ABAP Uploading"](#) for more details.
- SAP Allow ABAP Execute: set to 1, if ABAP code can be executed on this SAP System. See [Section 7.7.3, "Controlling ABAP Uploading"](#) for more details.

Note: The Test button for validating the SAP Connection and the FTP Connection definition is not supported.

- SAP Transport Layer Name: This transport layer is used by ODI to create any new transport requests. The default transport layer name is SAP. If the SAP system uses a different transport layer, this FlexField must be updated accordingly. Otherwise any TR creation will fail.

Except for Data Server Name, all the parameters that you provide while defining the SAP Data Server should be provided by the SAP Administrators. See [Gathering SAP Connection Information](#) for more information about these parameters.

7.3.2.2 Create the SAP ABAP Schema

To configure a SAP ABAP schema:

1. Create a Physical Schema under the SAP ABAP data server as described in "Creating a Physical Schema" in *Administering Oracle Data Integrator*. This schema does not require any specific configuration. Only one physical schema is required under a SAP ABAP data server.
2. Create a Logical Schema for this Physical Schema as described in "Creating a Logical Schema" in *Administering Oracle Data Integrator* in the appropriate context.

7.4 Setting up the Project

Setting up a project using SAP ERP features follows the standard procedure. See "Creating an Integration Project" of the *Developing Integration Projects with Oracle Data Integrator*.

Import the following KMs into your Oracle Data Integrator project:

- RKM SAP ERP
- RKM SAP ERP Connection Test
- LKM SAP ERP to Oracle (SQLLDR)
- LKM SAP ERP to SQL

In addition to these specific SAP ERP KMs, import the standard Oracle LKMs, IKMs, and CKMs to perform data extraction and data quality checks with an Oracle database. See "Oracle Database" in the *Connectivity and Knowledge Modules Guide for Oracle Data Integrator* for a list of available KMs.

7.5 Creating and Reverse-Engineering a Model

This section contains the following topics:

- [Creating a SAP ERP Model](#)
- [Reverse-Engineering a SAP ERP Model](#)

7.5.1 Creating a SAP ERP Model

Create an SAP ERP Model based on the SAP ABAP technology and on the SAP ABAP logical schema using the standard procedure, as described in "Creating a Model" of the *Developing Integration Projects with Oracle Data Integrator*.

7.5.2 Reverse-Engineering a SAP ERP Model

To perform a Customized Reverse-Engineering with the RKM SAP ERP, use the usual procedure, as described in "Reverse-engineering a Model" of the *Developing Integration Projects with Oracle Data Integrator*. This section details only the fields specific to SAP ERP:

1. In the Reverse Engineer tab of the SAP ERP Model, select the RKM SAP ERP.
2. For the RKM SAP ERP, set the USE_GUI KM option to true.
3. Save the model.
4. Click **Reverse-Engineer** in the Model Editor toolbar.
5. The Tree Metadata Browser appears after the session is started. Select the table(s) to reverse.
6. Click **Reverse-Engineer** in the Tree Metadata Browser window.

The reverse-engineering process returns the selected tables as datastores.

Note: If the reverse-engineering is executed on a run-time agent, the USE_GUI option should be set to false. This option should be used only when the customized reverse-engineering is started using the agent built-in the Studio.

7.6 Designing a Mapping

To create a mapping loading SAP ERP data into an Oracle staging area:

1. Create a mapping with source datastores from the SAP ERP Model. This mapping should have an Oracle target or use an Oracle schema as the Staging Area.
2. Create joins, filters, and map attributes for your mapping.
3. In the Physical diagram of the mapping, select the access point for the SAP ABAP source table(s). The Property Inspector opens for this object.
4. In the Loading Knowledge Module tab, select the LKM SAP ERP to Oracle (SQLDR)/LKM SAP ERP to SQL.

7.7 Considerations for SAP ERP Integration

This section includes the following topics:

- [File Transfer Configurations](#)
- [Execution Modes](#)
- [Controlling ABAP Uploading](#)
- [Managing ODI SAP Transport Requests](#)
- [SAP Packages and SAP Function Groups](#)
- [Log Files](#)

- [Limitations of the SAP ABAP Adapter](#)

7.7.1 File Transfer Configurations

The ODI SAP adapter extracts data using ABAP programs. For transferring the data from SAP system to the ODI agent the adapter supports two transfer modes and different configurations:

- [Transfer using a Shared Directory \(recommended\)](#)
- [FTP based Transfer](#)

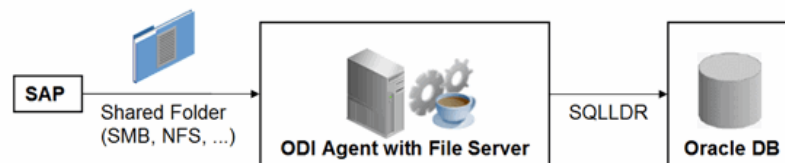
7.7.1.1 Transfer using a Shared Directory (recommended)

During the extraction process the ABAP programs write chunks of data into the data file in the shared folder. For better performances this shared folder should be located on the ODI agent machine. In this setup, SQL*Loader directly reads the data from this folder and loads it into database.

If the folder is not located on the ODI agent machine, then the ODI agent first needs to copy the file from the shared folder to the agent for loading the data using SQL*Loader/ JDBC-Connection in the next step.

Configuration 1: Shared Folder is physically located on the ODI Agent machine (recommended)

Figure 7–1 Configuration 1



This configuration is used, when `FTP_TRANSFER_METHOD = FSMOUNT_DIRECT`. In this configuration the following data movements are performed:

1. The ABAP program extracts chunks of `FETCH_BATCH SIZE` records and writes them into a file in the shared folder.
2. for LKM SAP ERP to Oracle (SQL*Loader): SQL*Loader reads locally the data file and loads the data into the Oracle staging area.

For LKM SAP ERP to SQL: The ODI File Driver reads locally the data file and inserts the data using JDBC into a non-Oracle staging area.

This configuration requires the following Topology settings:

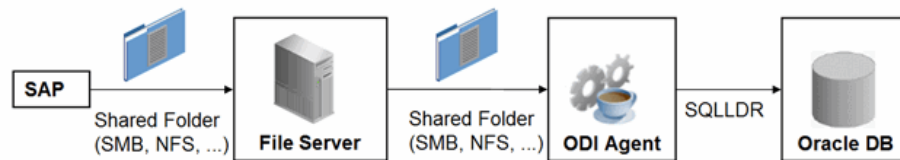
1. Create a File data server pointing to the File server/ODI Agent machine:
 - Host (Data Server): n/a
 - User: n/a
 - Password: n/a
2. Under this File data server create a physical schema representing the shared folder in the File host server. Specify the Data and Work Schemas as follows:
 - *Data Schema*: Path (UNC) of the shared folder used by the ABAP program to write extraction files.

- *Directory (Work Schema)*: Local path on the server's machine hosting the shared folder. This path is used by an agent and SQL*Loader installed on this machine to access the extraction files without passing via the shared folder.

Note: Temporary files such as ctl, bad, dsc will be created in a local temporary folder on the run-time agent. The default temporary directory is the system's temporary directory. On UNIX this is typically /tmp and on Windows c:\Documents and Settings\\Local Settings\Temp. This directory can be changed using the KM option TEMP_DIR.

Configuration 2: Shared folder is not physically located on the ODI Agent machine

Figure 7-2 Configuration 2



This configuration is used, when `FTP_TRANSFER_METHOD = FSMOUNT`. In this configuration the following data movements are performed:

1. The ABAP program extracts chunks of `FETCH_BATCH SIZE` records and writes them into a file in the shared folder.
2. The run-time agent copies the file into the directory given by `TEMP_DIR` option of the LKM.
3. for LKM SAP ERP to Oracle (SQL*Loader): SQL*Loader reads the data file from this `TEMP_DIR` and loads the data into the Oracle staging area.

For LKM SAP ERP to SQL: The ODI File Driver reads the data file from this `TEMP_DIR` and inserts the data using JDBC into a non-Oracle staging area.

This configuration requires the following Topology settings:

1. Create a File data server pointing to the file server into which the extraction file will be pushed from SAP and picked up from for SQL*Loader.

Set the parameters for this data server as follows:

- Host (Data Server): n/a
 - User: n/a
 - Password: n/a
2. In this File data server create a physical schema representing the shared folder. Specify the Data and Work Schemas as follows:
 - *Directory (Schema)*: Path (UNC) of the shared folder used by the ABAP program to write extraction files, and by the agent to copy the file.
 - *Directory (Work Schema)*: <undefined>. Leave this path blank, as data files are never accessed directly from the File server's file system.

Please note that data files will be copied to the run-time agent from the shared folder in a local temporary folder. The default temporary directory is the system's temporary directory. On UNIX this is typically /tmp and on Windows c:\Documents and Settings\

7.7.1.2 FTP based Transfer

During the extraction the ABAP program writes out data into a temporary file. This file is created in directory defined by the ODI DataServer FlexField SAP_TMP_DIR. See section [Section 7.2.3.4, "Gathering SAP Connection Information"](#) for more details about SAP_TMP_DIR.

At the end of the extraction process these ABAP programs will upload the data file to a FTP server. For better performances this FTP server should be located on the same machine as the run-time agent.

If the agent is not located on the same machine as the FTP server, it will download the file from the FTP server before loading it to the staging area using SQL*Loader / JDBC-Connection. This download operation is performed using FTP, SFTP or SCP.

Figure 7-3 Configuration 1: FTP Server is installed on an ODI Agent machine



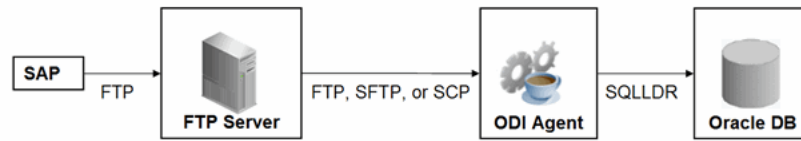
The configuration shown in [Figure 7-3](#) is used, when `FTP_TRANSFER_METHOD = NONE`. In this configuration the following data movements are performed:

1. The ABAP program extracts the data and uploads the data file to the FTP server.
2. For LKM SAP ERP to Oracle (SQL*Loader): SQL*Loader reads locally the data file and loads the data into the Oracle staging area.

For LKM SAP ERP to SQL: The ODI File Driver reads locally the data file and inserts the data using JDBC into a non-Oracle staging area.

This configuration requires the following Topology settings:

1. Create a File data server pointing to the FTP server:
 - Host (Data Server): FTP server host name or IP address.
 - User: Username to log into FTP server.
 - Password: Password for the user.
2. In this File data server create a physical schema representing the folder in the FTP host where the extraction file will be pushed. Specify the Data and Work Schemas as follows:
 - *Directory (Schema)*: Path on the FTP server for uploading SAP extraction files.
 - *Directory (Work Schema)*: Local path on the FTP server's machine containing the SAP extraction file. The agent and SQL*Loader/ODI Flat File Driver read the extraction files from this location.

Figure 7-4 Configuration 2: FTP Server is not installed on ODI Agent machine

The configuration shown in Figure 7-4 is used, when `FTP_TRANSFER_METHOD` is `FTP`, `SFTP` or `SCP`. In this configuration the following data movements are performed:

1. The ABAP program extracts the data and uploads the data file to the FTP server.
2. The ODI agent downloads the file from the FTP server into the directory given by KM Option `TEMP_DIR`.
3. For LKM SAP ERP to Oracle (SQLLDR): SQL*Loader reads the data file from this `TEMP_DIR` and loads the data into the Oracle staging area.

For LKM SAP ERP to SQL: The ODI File Driver reads the data file from this `TEMP_DIR` and inserts the data using JDBC into a non-Oracle staging area.

This configuration requires the following Topology settings:

1. Create a File data server pointing to the FTP server:
 - Host (Data Server): FTP server host name or IP address.
 - User: User name to log into FTP server.
 - Password: Password for the user.
2. In this File data server create a physical schema representing the folder in the FTP host where the extraction file will be pushed. Specify the Data and Work Schemas as follows:
 - *Directory (Schema)*: Path on the FTP server for uploading SAP extraction files.
 - *Directory (Work Schema)*: <undefined>; this path is left blank, as data files are never accessed directly from the FTP server's file system.

Considerations and Limitations:

The FTP based data transfer uses the widely spread (S)FTP file transfer and requires all data to be held in SAP's application server memory before transfer. Therefore the required memory per SAP session increases with the amount of data extracted and will set an upper limit to the data volume. This upper limit can be adjusted to a certain extend by increasing the sessions memory settings in SAP.

The required setup for the shared folder based configuration is slightly more complex, but it removes the need for all data to fit into SAP AS' memory and is therefore the recommended extraction method.

7.7.2 Execution Modes

Background Processing

By default the generated ABAP code will be deployed as an ABAP report. At execution this report is submitted to the SAP scheduler for background processing. The job status is then monitored by ODI.

The KM option `JOB_CLASS` defines the priority of the background Job. Valid values (corresponding to SAP `JOB_CLASS` settings) are as follows:

`JOB_CLASS = A` is Highest Priority

`JOB_CLASS = B` is Normal Priority

`JOB_CLASS = C` is Lowest Priority

Dialog mode processing

For backwards compatibility the KM option `BACKGROUND_PROCESSING` can be set to `false`. The generated ABAP code will then be deployed as an RFC. At execution this RFC is called by ODI to extract the data.

Dialog mode processing has been deprecated and is currently supported for backwards compatibility only. In the future releases dialog processing may be removed entirely in the future.

7.7.3 Controlling ABAP Uploading

During development ODI generates ABAP code and uploads it into the SAP development system with every mapping execution. This automatic code upload allows quick development cycles.

Once a Mapping or Package has been unit tested and is ready to be migrated out of the development environment, the generated SAP ABAP code has to be transported to the respective SAP system using SAP's CTS (Change and Control System) like any other SAP ABAP code. This is standard SAP practice. To facilitate this task, SAP transport requests are automatically created during upload into development. Please contact your SAP administrator for transporting generated ODI SAP ABAP programs.

In case you are working with distinct ODI repositories for dev, test and production, please make sure that your ODI scenario matches the ODI ABAP code of the respective SAP system. That is, you have to transport the SAP ABAP code using SAP CTS from your SAP development system to your SAP QA system *and* transport the ODI scenario (which has generated the transported ABAP code) from your ODI development repository to your ODI QA repository. Please see Chapter 13, "Working with Scenarios" of *Developing Integration Projects with Oracle Data Integrator* for details on how to transport ODI scenarios.

Once outside of development ODI should no longer upload ABAP code, as the ABAP code has been transported by SAP's CTS and such non-development systems usually do not allow ABAP uploading.

Even though uploading can be explicitly turned off by setting the LKM option `UPLOAD_ABAP_CODE` to `No`, it usually is turned off using the FlexField "SAP Allow ABAP Upload" defined on the SAP data server in ODI Topology: The ABAP code is only uploaded, if both the LKM option `UPLOAD_ABAP_CODE` and the Flexfield `SAP Allow ABAP Upload` are set to `Yes`. For disabling any upload into production systems it is sufficient to set the Flexfield "SAP Allow ABAP Upload" to 0 in Topology.

Tip: To configure a mapping that uploads the ABAP code in development but skips the upload in QA or production:

1. Set the KM option `UPLOAD_ABAP_CODE` set to `Yes` in all mappings
2. Configure the SAP data servers in the Topology as follows:
 - Set the Flexfield *SAP Allow ABAP Upload* to 1 for all SAP development systems
 - Set the Flexfield *SAP Allow ABAP Upload* to 0 for all other SAP systems

Note: Before starting the extraction process, ODI verifies that the mapping/scenario matches the code installed in SAP. If there is a discrepancy - for example, if the scenario was modified but the ABAP code was not re-uploaded - an exception is thrown.

In some situations it may be desirable to install the Mapping's ABAP extraction code and not to extract any data, such as for an automated installation. In this case all mappings can be linked inside a package with the KM option `EXECUTE_ABAP_CODE` set to `False` in every mapping. Executing this package will then install all ABAP code, but will not perform any execution.

To avoid the modification of all mappings (setting `EXECUTE_ABAP_CODE` to `False` as described above), you can instead disable all SAP ABAP executions by using the FlexField **SAP Allow ABAP Execute** on the ODI DataServer. If this FlexField is *disabled*, the ABAP code is not executed even if the KM option `EXECUTE_ABAP_CODE` is set to `True`.

Manual Upload

In some cases automatic upload may not be allowed even in development systems. For such situations the KM option `MANUAL_ABAP_CODE_UPLOAD` allows manual uploads. If set to `true`, ODI will create a text file containing the generated ABAP code. By default, the name of the text file is similar to `ZODI_<Mapping Id>_<SourceSet Id>.ABAP`. This code is handed over to the SAP administrator, who will install it. Once installed the ODI mapping can be executed with `MANUAL_ABAP_CODE_UPLOAD` and `UPLOAD_ABAP_CODE` both set back to `false`.

7.7.4 Managing ODI SAP Transport Requests

During development, ABAP code is uploaded to the SAP system with every mapping execution. More precisely:

An ODI mapping extracting SAP data generates one or several ABAP extraction programs (e.g. when join location is set to staging and consequently two extraction jobs are created). By default all ABAP extraction programs of one mapping are assigned to one SAP function group. The ABAP extraction programs for a different mapping will be assigned to a different SAP function group. The default function group name is similar to `ZODI_FGR_<Mapping Id>`.

During upload a SAP CTS transport request is created for each ODI Mapping (for each SAP function group). This allows granular deployment of the generated ODI ABAP extraction programs via SAP CTS.

Grouping (Background processing)

When the ABAP code of multiple ODI Mappings should be grouped into a single transport request for more coarse-grained deployment control, the following steps are needed:

- Set KM option `SAP_REPORT_NAME_PREFIX` for a common prefix for all ABAP reports. For example, `ZODI_DWH_SALES_` on all mappings.
- Choose a KM option `ABAP_PROGRAM_NAME` for every individual mapping. For example, `LOAD01`, `LOAD02`, etc.

These sample settings would result in a single transport request containing the ABAP reports called `ZODI_DWH_SALES_LOAD01`, `ZODI_DWH_SALES_LOAD02`, etc.

Grouping (Dialog mode processing, deprecated)

When the ABAP code of multiple ODI Mappings should be grouped into a single transport request for more coarse-grained deployment control, the KM option `SAP_FUNCTION_GROUP` for all LKMs in these mappings can be set to a user defined value, e.g. `ZODI_FGR_DWH_SALES`. This then leads to ODI generating all ABAP extraction programs into the same SAP function group which is then attached to a single transport request. For valid function group names at your site please contact your SAP administrator.

Tip: The name of the generated ABAP extraction programs is by default similar to `ZODI_<Mapping Id>_<SourceSet Id>`. This ensures convenient development due to unique program names. While the MappingId never changes, certain changes to an ODI mapping can cause the SourceSetId to change and consequently cause the respective extraction program name to change. Therefore it is recommended to use user-defined program names, once development stabilizes. ABAP program names can be set by defining a value for LKM option `ABAP_PROGRAM_NAME`, e.g. `ZODI_DWH_SALES_DATA01`. Please contact your SAP administrator for applicable naming conventions.

Transport Request Description

When ODI creates a transport request, it will set the transport request description to the text provided in KM option `SAP_TRANSPORT_REQUEST_DESC` and applies to function group defined in KM option `FUNCTION_GROUP`.

By default the ODI Step name (which is usually the mapping name) will be used.

Code generation expressions like `ODI:<%=odiRef.getPackage("PACKAGE_NAME")%>` may be useful when grouping several mappings into one SAP function group/ SAP transport request.

7.7.5 SAP Packages and SAP Function Groups**SAP Packages**

All SAP objects installed by the ODI SAP Adapter are grouped into SAP packages:

- `ZODI_RKM_PCKG` contains any RKM related objects. These objects are used during development phase.
- `ZODI_LKM_PCKG` contains any extraction programs generated by any SAP LKM.

If requested by the SAP administrator, these default names can be overwritten using the KM options `SAP_PACKAGE_NAME_ODI_DEV` and `SAP_PACKAGE_NAME_ODI_PROD`. These

values must be set during the first-time installation. Later changes require a reinstallation of the ODI SAP Adatper with a prior uninstallation.

Please note that LKM option `SAP_PACKAGE_NAME` must always use the same value given during first-time RKM installation. This means that when non-default values are used during first-time RKM installation, **all** mappings must set the LKM option `SAP_PACKAGE_NAME` to the non-default value.

SAP Function Groups

All SAP function modules are grouped into SAP function groups:

- `ZODI_FGR` contains any function modules needed during development, e.g. for retrieving metadata. These function modules are installed by the RKM during first-time installation.

The default name can be overwritten during first-time RKM installation using RKM option `SAP_FUNCTION_GROUP_ODI_DEV`.

- `ZODI_FGR_PROD` contains any function modules needed at runtime in production, e.g. for monitoring ABAP report execution status. These function modules are installed by the RKM during first-time installation.

The default name can be overwritten during first-time RKM installation using RKM option `SAP_FUNCTION_GROUP_ODI_PROD`.

- `ZODI_FGR_PROD_...` contains any data extraction function modules. Such function modules are generated by the LKM when using deprecated dialog mode (`BACKGROUND_PROCESSING=false`).

By default every mapping uses its own function group. The default values can be overwritten using the LKM option `SAP_FUNCTION_GROUP`, which is independent of the two function group names mentioned above.

See [Managing ODI SAP Transport Requests](#) for more information.

7.7.6 Log Files

During RKM and LKM execution many messages are logged in ODI's logging facility. These messages in the ODI log files and other log files may contain valuable details for troubleshooting. See section "Runtime Logging for ODI components" of the *Administering Oracle Data Integrator* for details about the ODI's logging facility.

[Table 7-2](#) describes the different log files and their usage:

Table 7-2 Log Files

Default Log File Name	KM / Phase	Content
<ODI Logs>	RKM	Execution Log of metadata retrieval
<ODI Logs>	RKM	Information about first time installation of SAP RFC for RKM
<ODI Logs>	LKM - Generation Time	Information about code generation for ABAP extractor
<ODI Logs>	LKM - Runtime	Information about installation of ABAP extractor
<System Temp Dir or local FTP dir>/ ZODI_<Mapping Id>_<SrcSet>_<Context>.log	LKM - Runtime	SQL*Loader log file

Table 7–2 (Cont.) Log Files

Default Log File Name	KM / Phase	Content
<System Temp Dir or local FTP dir>/ ZODI_<Mapping Id>_<SrcSet>_<Context>.out	LKM - Runtime	OS std output during SQL*Loader execution, may contain information, e.g. when SQL*Loader is not installed
<System Temp Dir or local FTP dir>/ ZODI_<Mapping Id>_<SrcSet>_<Context>.err	LKM - Runtime	OS error output during SQL*Loader execution, may contain information, e.g. when SQL*Loader is not installed

7.7.7 Limitations of the SAP ABAP Adapter

The SAP ABAP adapter has the following limitations:

- The **Test** button for validating SAP Connection definition in ODI's Topology manager is not supported.
- The SAP table type (Transparent, Pool, and Cluster) cannot be changed after a table has been reverse-engineered.
- The SAP ABAP KMs only support Ordered Joins.
- Full Outer join and Right outer joins are not supported.
- In one-to-many relationships, the first table of a join needs to be the one-table, for example when joining MARA and MARC, MARA needs to be the first table in the join.
- Mapping expression executed on the source must not contain any transformations.

Additional Information for SAP ABAP BW Adapter

This appendix describes the privileges that are required for connecting to SAP System, and how you can test the connection outside of ODI using a standalone java utility.

This appendix includes the following sections:

- [Section A.1, "SAP ABAP BW Required Privileges"](#)
- [Section A.2, "SAP Stand-Alone Connection Test"](#)

For more information about the SAP BW KMs see *Connectivity and Knowledge Modules Guide for Oracle Data Integrator*.

A.1 SAP ABAP BW Required Privileges

For connecting to the SAP system a SAP dialog user is required for RKM installation and LKM execution during development. A SAP RFC user is *not* sufficient.

This user has to have a developer license key. License keys can be retrieved from <http://service.sap.com>. Any execution attempts without this developer license key will lead to failure and may cause the need for clean-up operations.

Later for RKM execution and LKM execution in production a SAP RFC user is sufficient.

These SAP user types can be used for the following operations:

Table A-1 Required SAP User Types

Operation	Required SAP User Type
RKM: Setup	SAP Dialog user
RKM: Reverse Engineering	SAP Dialog user or SAP RFC user
LKM: Development	SAP Dialog user or SAP RFC user
LKM: Production	SAP Dialog user or SAP RFC user
Testing ODI Generated Extraction RFCs	SAP Dialog user or SAP RFC user

The following tables list the privileges required for using SAP BW Knowledge Modules:

- [Authorizations Required for RKM SAP BW Setup](#)
- [Authorizations Required for RKM SAP BW Execution](#)

- [Authorizations Required for LKM SAP BW Execution](#)
- [Authorizations Required for LKM SAP BW Execution in Production](#)
- [Authorizations Required for LKM SAP BW Execution as Background Process](#)
- [Authorizations Required for LKM SAP BW Execution as Background Process in Production](#)

A.1.1 Important points to consider

Consider the following points while configuring the SAP privileges:

- **S_DATASET** is an authorization object that controls access to physical file, so you need to provide access to SAP directories & folder mounted path. The values provided here are sample paths, you need to provide your landscape directories path or provide ***(all)**.
- **NR** or **(NR)** means **NOT REQUIRED**.

A.1.2 Authorizations Required for RKM SAP BW Setup

The following authorizations are required for **RKM SAP BW** setup.

Note: Developer Key is required in this authorization.

Object	Field Name	Value (BW 3.5)	Value (BI 7.0)
S_RFC	ACTVT	16	16
	RFC_NAME	RFC1, RSAB, RSB3RD, SDIFRUNTIME, SUTL, SYST, SYSU, ZODI_ FGR	RFC1, RSAB, RSB3RD, SDIFRUNTIME, SUTL, SYST, ZODI_*
	RFC_TYPE	FUGR	FUGR
S_TCODE	TCD	SU53, SE38	SU53, SE38
	S_ADMI_FCD		MEMO
S_CTS_ADMI	CTS_ADMFCT	TABL	TABL
S_DATASET	ACTVT	34	06, 33, 34
	FILENAME	*	*
	PROGRAM	SAPLSTRF	SAPLSLOG, SAPLSTRF, SAPLZODI*
S_TABU_DIS	ACTVT		03
	DICBERCLS		*
S_DEVELOP	ACTVT	01, 02, 03, 06	01, 02, 03, 06, 07, 16
	DEVCLAS	\$TMP, ZODI_LKM_ PCKG, ZODI_RKM_ PCKG	\$TMP, RSS, ZODI_ LKM_PCKG, ZODI_ RKM_PCKG

Object	Field Name	Value (BW 3.5)	Value (BI 7.0)
	OBJNAME	Z\$\$\$XRFC, ZODI_FGR, ZODI_FGR_PROD, ZODI_LKM_PCKG, ZODI_RKM_PCKG	Z\$\$\$XRFC, RSSB, ZODI_FGR, ZODI_FGR_PROD, ZODI_LKM_PCKG, ZODI_RKM_PCKG
	OBJTYPE	DEVC, FUGR, PROG	DEVC, FUGR, PROG
	P_GROUP		*
S_TRANSPRT	ACTVT	01,02,03,06	01,02,03
	TTYPE	DTRA, TASK	DTRA, TASK
S_RS_ADMWB	ACTVT	03,16	03,16
	RSADMWBOBJ	INFOBJECT	INFOBJECT
S_RS_ICUBE	ACTVT	03	03
	RSICUBEBOBJ	*	*
	RSINFOAREA	*	*
	RSINFOCUBE	*	*
S_RS_IOBJ	ACTVT	03	03
	RSIOBJ	*	*
	RSIOBJCAT	*	*
	RSIOBJPART	DEFINITION	DEFINITION
S_RS_ODSO	ACTVT	03	03
	RSINFOAREA	*	*
	RSODSOBJ	*	*
	RSODSPART	DATA,DEFINITION	DATA,DEFINITION

A.1.3 Authorizations Required for RKM SAP BW Execution

The following authorizations are required for RKM SAP BW execution.

Object	Field Name	Value (BW 3.5)	Value (BI 7.0)
S_RFC	ACTVT	16	16
	RFC_NAME	RFC1, RSB3RD, SDIFRUNTIME, SYST, ZODI_FGR, RSAB	RFC1, RSB3RD, SDIFRUNTIME, SYST, ZODI_*
	RFC_TYPE	FUGR	FUGR
S_TCODE	TCD	SU53	SU53, SE38
S_TABU_DIS	ACTVT	03	03
	DICBERCLS	*	MA, SC
S_RS_ADMWB	ACTVT	03,16	03
	RSADMWBOBJ	INFOBJECT	INFOBJECT
S_RS_ICUBE	ACTVT	03	03
	RSICUBEBOBJ	*	*
	RSINFOAREA	*	*

Object	Field Name	Value (BW 3.5)	Value (BI 7.0)
	RSINFOCUBE	*	*
S_RS_IOBJ	ACTVT	03	03
	RSIOBJ	*	0*
	RSIOBJCAT	*	0*
	RSIOBJPART	DEFINITION	DEFINITION
S_RS_ODSO	ACTVT	03	03
	RSINFOAREA	*	0*, T*, Z*
	RSODSOBJ	*	0*, U*, Z*
	RSODSPART	DATA, DEFINITION	DATA, DEFINITION

A.1.4 Authorizations Required for LKM SAP BW Execution

The following authorizations are required for LKM SAP BW execution.

Note: Developer Key is required in this authorization.

Object	Field Name	Value (BW 3.5)	Value (BI 7.0)
S_RFC	ACTVT	16	16
	RFC_NAME	RFC1, RSAB, SDIF, SYSU, SDIFRUNTIME, SYST, ZODI_FGR, ZODI_FGR_PROD*	RFC1, SDIFRUNTIME, SYST, ZODI_FGR, ZODI_FGR_PROD*
	RFC_TYPE	FUGR	FUGR
S_TCODE	TCOD	SE38, SU53	SE38, SU53
S_CTS_ADMI	CTS_ADMFCT	TABL	TABL
S_DATASET	ACTVT	06, 33, 34	06, 33, 34
	FILENAME	*\\DEL-7TMK2BS\odi agentbox*, \\DEL-7TMK2BS\odia gentbox	\\10.30.0.201\ODI_SAPTEST, \\10.30.0.201\ODI_SAPTEST\ZODI_233030*
	PROGRAM	SAPLSLOG, SAPLSTRF, SAPLZODI*	SAPLSLOG, SAPLSTRF, SAPLZODI*
S_DEVELOP	ACTVT	01, 02, 03, 06	01, 02, 03, 06
	DEVCLASS		
	OBJNAME	ZODI_FGR_PROD*	ZODI_FGR_PROD*
	OBJTYPE	FUGR, PROG	FUGR, PROG
	P_GROUP		
S_PROGRAM	P_ACTION		EDIT, SUBMIT, VARIANT
	P_GROUP		
S_TRANSPRT	ACTVT	01	01

Object	Field Name	Value (BW 3.5)	Value (BI 7.0)
	TTYPE	DTRA, TASK	DTRA, TASK
S_RS_AUTH	BIAUTH		0*
S_RS_ICUBE	ACTVT	03	
	RSICUBE OBJ	DATA, DEFINITION	
	RSINFOAREA	NODESNOTCONNECTED	
	RSINFOCUBE	0*, Z*	

A.1.5 Authorizations Required for LKM SAP BW Execution in Production

The following authorizations are required for LKM SAP BW execution in production.

Object	Field Name	Value (BW 3.5)	Value (BI 7.0)
S_RFC	ACTVT	16	16
	RFC_NAME	RFC1, SDIFRUNTIME, SYST, ZODI_FGR_PROD*	RFC1, SDIFRUNTIME, SYST, ZODI_FRG*, ZODI_FGR_PROD*
	RFC_TYPE	FUGR	FUGR
S_TCODE	TCD	SU53	SU53
S_DATASET	ACTVT	06, 33, 34	06, 33, 34
	FILENAME	*\\DEL-7TMK2BS\odi agentbox*, \\DEL-7TMK2BS\odia gentbox	\\10.30.0.201\ODI_SAPTEST, \\10.30.0.201\ODI_SAPTEST\ZODI_233030*
	PROGRAM	SAPLSLOG, SAPLSTRF, SAPLZODI*	SAPLSLOG, SAPLSTRF, SAPLZODI*
S_RS_AUTH	BIAUTH		0*
S_RS_ICUBE	ACTVT	03	03
	RSICUBE OBJ	DATA, DEFINITION	DEFINITION
	RSINFOARE	NODESNOTCONNECTED	
	RSINFOCUBE	0*, Z*	
S_RS_IOBJ	ACTVT		03
	RSIOBJ		
	RSIOBJCAT		
	RSIOBJPART		DEFINITION
S_RS_ODSO	ACTVT		03
	RSINFOAREA		
	RSODSOBJ		
	RSODSPART		DEFINITION

A.1.6 Authorizations Required for LKM SAP BW Execution as Background Process

The following authorizations are required for LKM SAP BW execution as background process.

Note: Developer Key is required in this authorization.

Object	Field Name	Value (BW 3.5)	Value (BI 7.0)
S_RFC	ACTVT	16	16
	RFC_NAME	RFC1, SDIFRUNTIME, RSAB, SYSU, SLST, SYST, ZODI_FGR, ZODI_FGR_PROD*	RFC1, SDIFRUNTIME, SLST, SYST, ZODI_FGR, ZODI_FGR_PROD*
	RFC_TYPE	FUGR	FUGR
S_TCODE	TCD	SE38, SU53	SE38, SU53
S_BTCH_ADM	BTCADMIN	Y	Y
S_BTCH_JOB	JOBACTION	RELE	RELE
	JOBGROUP	'	'
S_CTS_ADMI	CTS_ADMFCT	TABL	TABL
S_DATASET	ACTVT	06, 33, 34	06, 33, 34
	FILENAME	*\\DEL-7TMK2BS\odi agentbox*, \\DEL-7TMK2BS\odia gentbox, \\DELRC6\sapmnt*	\\10.30.0.201\ODI_SAPTEST, \\10.30.0.201\ODI_SAPTEST\ZODI_233030*
	PROGRAM	SAPLSLOG, SAPLSTRF, SAPLZODI*, ZODI*	SAPLSLOG, SAPLSTRF, SAPLZODI*, ZODI*
S_DEVELOP	ACTVT	01	01, 02, 03, 06, 16
	DEVCLASS		
	OBJNAME	ZODI_FGR_PROD*	ZODI_FGR_PROD*
	OBJTYPE	FUGR	FUGR, PROG
	P_GROUP		
S_PROGRAM	P_ACTION		EDIT, SUMMIT, VARIAN T
	P_GROUP		
S_TRANSPRT	ACTVT	01, 02, 03	01, 02
	TTYPE	DTRA, TASK	DTRA, TASK
S_RS_AUTH	BIAUTH		0*
S_RS_ICUBE	ACTVT	03, 06	
	RSICUBEOBJ	DATA, DEFINITION	
	RSINFOAREA	NODESNOTCONNECTED	
	RSINFOCUBE	0*, Z*	
S_APP_LOG	ACTVT	03, 06	03, 06

Object	Field Name	Value (BW 3.5)	Value (BI 7.0)
	ALG_OBJ	ZODI_APPOBJ	ZODI_APPOBJ
	ALG_SUBOBJ	ZODI_SUBOBJ	ZODI_SUBOBJ

A.1.7 Authorizations Required for LKM SAP BW Execution as Background Process in Production

The following authorizations are required for LKM SAP BW execution as background process in a SAP production environment.

Object	Field Name	Value (BW 3.5)	Value (BI 7.0)
S_RFC	ACTVT	16	16
	RFC_NAME	RFC1, SDIFRUNTIME, SYSU, SLST, SYST, ZODI*	RFC1, SDIFRUNTIME, SLST, SYST, ZODI*
	RFC_TYPE	FUGR	FUGR
S_TCODE	TCD	SU53	SU53
S_BTCH_ADM	BTCADMIN	Y	Y
S_BTCH_JOB	JOBACTION	RELE	RELE
	JOBGROUP	' '	' '
S_DATASET	ACTVT	06, 33, 34	06, 33, 34
	FILENAME	\\10.30.0.201\ODI_SAPTEST*, \\DEL-7TMK2BS\odia gentbox*, \\DELRC6\sapmnt*	\\10.30.0.201\ODI_SAPTEST*, \\DEL-7TMK2BS\odia gentbox*, \\DELRC6\sapmnt*
	PROGRAM	SAPLSTRF, ZODI*	ZODI*
S_DEVELOP	ACTVT		01, 02, 03, 06, 07, 16
	DEVCLASS		
	OBJNAME		S_TCODE
	OBJTYPE		SUSO
	P_GROUP		
S_RS_AUTH	BIAUTH		0*
S_RS_ICUBE	ACTVT	03	03
	RSICUBE0B	DATA, DEFINITION	DATA, DEFINITION
	RSINFOAREA	NODESNOTCONNECTED	0*, Z*
	RSINFOCUBE	0*, Z*	0*, Z*
S_APPL_LOG	ACTVT		03, 06
	ALG_OBJECT		ZODI_APPOBJ
	ALG_SUBOBJ		ZODI_SUBOBJ

A.2 SAP Stand-Alone Connection Test

In addition to the Connection Testing described in the *Getting Started with SAP ABAP BW Adapter for Oracle Data Integrator*, a test can be performed outside of ODI using a standalone java utility. This test is the same for SAP ERP and SAP BW.

See [Section B.2, "SAP Connection Test"](#) for more information.

A.2.1 SAP Stand-Alone Connection Test

In addition to the Connection Testing, a test can be performed outside of ODI using a standalone java utility. This utility is available with the Standalone agent.

To use a standalone java utility to test the connection:

1. Open a command window.
2. Go to the <ODI_HOME>/odi/sdk/lib directory.
3. Make sure that JAVA_HOME points to a supported JVM.
4. Make sure that you have installed SAP Java Connector and that the sapjco3.jar and the sapjco3 library are in the <ODI_HOME>/odi/sdk/lib directory.
5. Launch the utility using the following command:

On Windows:

```
java -cp sapjco3.jar;odi-sap.jar oracle.odi.sap.km.test.JCoTest
```

On Linux/UNIX:

```
java -cp sapjco3.jar:odi-sap.jar oracle.odi.sap.km.test.JCoTest
```

This command generates a ODI_SAP_CON_POOL.jcoDestination file in the folder <ODI_HOME>/odi/sdk/lib folder.

For more information, see "Appendix C, SAP Stand-Alone Connection Test" in the *Getting Started with SAP ABAP BW Adapter for Oracle Data Integrator*.

6. Use a text editor to open the ODI_SAP_CON_POOL.jcoDestination file. This file should look as follows:


```
#for tests only!
jco.client.lang=EN
jco.destination.peak_limit=10
jco.client.client=800
jco.client.passwd=<SAP Password>
jco.client.user=<SAP User>
jco.client.sysnr=00
jco.destination.pool_capacity=5
jco.client.ashost=<SAP Application Server>
```
7. Enter you SAP connection information, which you have received from your SAP administrator.
8. Launch the utility using the same command. The utility uses the file that you have edited, and outputs the test results or the possible issues.

In addition to just testing the SAP connection, the utility will also validate the existence of certain Function Modules required for the RKM. These are installed during first execution of the RKM (UPLOAD_ABAP_CODE and UPLOAD_ABAP_BASE set to true).

9. Delete the ODI_SAP_CON_POOL.jcoDestination file after execution, as it contains the SAP login credentials.

Additional Information for SAP ABAP ERP Adapter

This appendix describes the privileges that are required for connecting to SAP System, how you can test the connection outside of ODI using a standalone java utility, and how to uninstall SAP components.

This appendix includes the following sections:

- [Section B.1, "SAP ABAP ERP Required Privileges"](#)
- [Section B.2, "SAP Connection Test"](#)
- [Section B.3, "SAP Stand-Alone Connection Test"](#)

For more information about the SAP ERP KMs see *Connectivity and Knowledge Modules Guide for Oracle Data Integrator*.

B.1 SAP ABAP ERP Required Privileges

An SAP dialog user is required for connecting to the SAP system, for the RKM installation and for the LKM execution during development. A SAP RFC user is *not* sufficient.

During development, this user must have a developer license key. License keys can be retrieved from <http://service.sap.com>. Any execution attempts without this developer license key will lead to failure and may cause the need for clean-up operations.

For RKM and LKM execution in production, an SAP RFC user is sufficient.

These SAP user types can be used for the following operations:

Table B-1 Required SAP User Types

Operation	Required SAP User Type
RKM: Setup	SAP Dialog user
RKM: Reverse Engineering	SAP Dialog user or SAP RFC user
LKM: Development	SAP Dialog user or SAP RFC user
LKM: Production	SAP Dialog user or SAP RFC user

The following tables list the privileges required for using SAP ABAP Knowledge Modules:

- [Authorizations Required for RKM SAP ERP Setup](#)
- [Authorizations Required for RKM SAP ERP Execution](#)

- [Authorizations Required for LKM SAP ERP Execution](#)
- [Authorizations Required for LKM SAP ERP Execution in Production](#)
- [Authorizations Required for LKM SAP ERP Execution as Background Process](#)
- [Authorizations Required for LKM SAP ERP Execution as Background Process in Production](#)

B.1.1 Important points to consider

Consider the following points while configuring the SAP privileges:

- **S_DATASET** is an authorization object that controls access to physical file, so you need to provide access to SAP directories & folder mounted path. The values provided here are sample paths, you need to provide your landscape directories path or provide ***(all)**.
- **NR** or **(NR)** means **NOT REQUIRED**.

B.1.2 Authorizations Required for RKM SAP ERP Setup

The following SAP authorizations are required for **RKM SAP ERP** setup.

Note: Developer key is required in this authorization.

Object	Field Name	Value (4.6c)	Value (4.7)	Value (ECC 5)	Value (ECC 6)
S_RFC	ACTVT	16	16	16	16
	RFC_NAME	RFC1, SDIFRUNTIME, SUTL, SYST, ZODI_FGR, SDIF	RFC1, SDIFRUNTIME, SUTL, SYST, ZODI_FGR	RFC1, SDIFRUNTIME, SUTL, SYST, ZODI_FGR	RFC1, SDIF, SDIFRUNTIME, SUTL, SYST, SYSU, ZODI_ FGR
	RFC_TYPE	FUGR	FUGR	FUGR	FUGR
S_TCODE	TCD	SU53, SE38	SU53, SE38	SU53, SE38	SU53, SE38
S_ADMI_FCD	S_ADMI_FCD	MEMO, SM02	MEMO	MEMO	
S_CTS_ADMI	CTS_ADMFCT	TABL	TABL	TABL	TABL
S_DATASET	ACTVT	34	34	34	34
	FILENAME	\\HYPTTEST01\ sapmnt*	G:\usr*, \\10.30.0.20 1\ODI_ SAPTEST*, \\10.30.32.3 0*	D:\usr*, \\10.30.0.20 1\ODI_ SAPTEST*, \\10.30.32.4 2*	*
	PROGRAM	SAPLSLOG, SAPLSTRF	SAPLSLOG, SAPLSTRF	SAPLSLOG, SAPLSTRF, SAPLZODI*, ZODI*	SAPLSLOG, SAPLSTRF
S_TABU_DIS	ACTVT	03	03	03	03
	DICBERCLS	*	*	*	*
S_DEVELOP	ACTVT	01, 02, 03	01, 02, 03	01, 02, 03	01, 02, 03

Object	Field Name	Value (4.6c)	Value (4.7)	Value (ECC 5)	Value (ECC 6)
	DEVCLAS	\$TMP, ZODI_ LKM_PCKG, ZODI_RKM_ PCKG	\$TMP, ZODI_ LKM_PCKG, ZODI_RKM_ PCKG	\$TMP, ZODI_ LKM_PCKG, ZODI_RKM_ PCKG	\$TMP, ZODI_ LKM_PCKG, ZODI_RKM_ PCKG
	OBJNAME	Z\$\$\$XRFC, ZODI_FGR, ZODI_FGR_ PROD, ZODI_ LKM_PCKG, ZODI_RKM_ PCKG	Z\$\$\$XRFC, ZODI_FGR, ZODI_FGR_ PROD, ZODI_ LKM_PCKG, ZODI_RKM_ PCKG	Z\$\$\$XRFC, ZANU*, ZODI_ FGR, ZODI_ FGR_PROD, ZODI_LKM_ PCKG, ZODI_ RKM_PCKG	Z\$\$\$XRFC, ZODI_FGR, ZODI_FGR_ PROD, ZODI_ LKM_PCKG, ZODI_RKM_ PCKG, ZODI_ MSG
	OBJTYPE	DEVC, FUGR, PR OG	DEVC, FUGR, PR OG, MSAG	DEVC, FUGR, PR OG	DEVC, FUGR, PR OG, MSAG
	P_GROUP				
S_TRANSPRT	ACTVT	01, 03	01	01, 03	01
	TTYPE	DTRA, TASK	DTRA, TASK	DTRA, TASK	DTRA, TASK

B.1.3 Authorizations Required for RKM SAP ERP Execution

The following SAP authorizations are required for RKM SAP ERP execution.

Object	Field Name	Value (4.6c)	Value (4.7)	Value (ECC 5)	Value (ECC 6)
S_RFC	ACTVT	16	16	16	16
	RFC_NAME	RFC1, SDIF, SY ST, ZODI_ FGR, SDIFRUNT IME	RFC1, SDIFRUN TIME, SYST, ZO DI_FGR	RFC1, SDIFRUN TIME, SYST, ZO DI_FGR	RFC1, SDIF, SDIFRUNTIME, SYST, ZODI_ FGR
	RFC_TYPE	FUGR	FUGR	FUGR	FUGR
S_TCODE	TCD	SU53	SU53	SU53	SU53
S_TABU_DIS	ACTVT	03	03	03	03
	DICBERCLS	*	*	*	*

B.1.4 Authorizations Required for LKM SAP ERP Execution

The following SAP authorizations are required for LKM SAP ERP execution.

Note: Developer Key is required in this authorization.

Object	Field Name	Value (4.6c)	Value (4.7)	Value (ECC 5)	Value (ECC 6)
S_RFC	ACTVT	16	16	16	16
	RFC_NAME	RFC1, SDIF, SYST, ZODI_ FGR, ZODI_ FGR_PROD*	RFC1, SDIFRUN TIME, SYST, ZODI_FGR, ZODI_FGR_ PROD*	RFC1, SDIFRUN TIME, SYST, ZODI_FGR, ZODI_FGR_ PROD*	RFC1, SDIFRUN TIME, SYST, ZODI_FGR, ZODI_FGR_ PROD*
	RFC_TYPE	FUGR	FUGR	FUGR	FUGR
S_TCODE	TCD	SU53	SE38, SU53	SE38, SU53	SU53

Object	Field Name	Value (4.6c)	Value (4.7)	Value (ECC 5)	Value (ECC 6)
S_DATASET	ACTVT	06,33,34	06,33,34	06,33,34	06,33,34
	FILENAME	\\10.30.0.20 1\ODI_ SAPTEST\ZODI *, \\HYPTTEST01\ sapmnt*	*\\10.30.0.2 01\ODI_ SAPTEST*, \\10.30.0.20 1\ODI_ SAPTEST\ZODI _233030*	*\\10.30.0.2 01\ODI_ SAPTEST*, \\10.30.0.20 1\ODI_ SAPTEST\ZODI _233030*	*\\DEL-7TMK2 BS\odiagentb ox*, \\DEL-7TMK2B S\odiagentbo x
	PROGRAM	SAPLSLOG, SAPLSTRF, SAPLZODI*, ZODI*	SAPLSLOG, SAPLSTRF, SAPLZODI	SAPLSLOG, SAPLSTRF, SAPLZODI*, ZODI*	SAPLSLOG, SAPLSTRF, SAPLZODI*, ZODI*
S_DEVELOP	ACTVT	01	01,02,03,06	01,02,03,06	01
	DEVCLASS		ZODI_LKM_ PCKG		
	OBJNAME	ZODI_FGR_ PROD*	ZODI_FGR_ PROD*	ZODI_FGR_ PROD*	ZODI_FGR_ PROD*
	OBJTYPE	FUGR	FUGR, PROG	FUGR	FUGR
S_PROGRAM	P_ACTION		EDIT, SUBMIT, VARIANT	EDIT, SUBMIT, VARIANT	
	P_GROUP				
S_CTS_ADMI	CTS_ADMFCT		TABL		
S_TRANSPRT	ACTVT	01	01	01	01
	TTYPE	DTRA, TASK	DTRA, TASK	DTRA, TASK	DTRA, TASK

B.1.5 Authorizations Required for LKM SAP ERP Execution in Production

The following SAP authorizations are required for LKM SAP ERP execution in the SAP production environment.

Object	Field Name	Value (4.6c)	Value (4.7)	Value (ECC 5)	Value (ECC 6)
S_RFC	ACTVT	16	16	16	16
	RFC_NAME	SDIFRUNTIME, SYST, ZODI*	RFC1, SDIFRUNTIME, SYST, ZODI_ FGR_PROD*	RFC1, SDIFRUNTIME, SYST, ZODI_ FGR_PROD*	RFC1, SDIFRUNTIME, SYST, ZODI_ FGR_PROD*
	RFC_TYPE	FUGR	FUGR	FUGR	FUGR
S_TCODE	TCD	SU53	SU53	SU53	SU53
S_DATASET	ACTVT		06,33,34	06,33,34	06,33,34
	FILENAME		*\\DEL-7TMK2 BS\odiagentb ox*, \\DEL-7TMK2B S\odiagentbo x	*\\DEL-7TMK2 BS\odiagentb ox*, \\DEL-7TMK2B S\odiagentbo x	*\\DEL-7TMK2 BS\odiagentb ox*, \\DEL-7TMK2B S\odiagentbo x

Object	Field Name	Value (4.6c)	Value (4.7)	Value (ECC 5)	Value (ECC 6)
	PROGRAM		SAPLSLOG, SAPLSTRF, SAPLZODI	SAPLSLOG, SAPLSTRF, SAPLZODI*	SAPLSLOG, SAPLSTRF, SAPLZODI*,ZO DI*

B.1.6 Authorizations Required for LKM SAP ERP Execution as Background Process

The following SAP authorizations are required for executing LKM SAP ERP as background process.

Object	Field Name	Value (4.6c)	Value (4.7)	Value (ECC 5)	Value (ECC 6)
S_RFC	ACTVT	16	16	16	16
	RFC_NAME	RFC1,SDIF, SLST,SYST, ZODI*,ZODI_ FGR,ZODI_ FGR_PROD*	RFC1, SDIFRUNTIME, SLST,SYST, ZODI_FGR, ZODI_FGR_ PROD*	RFC1, SDIFRUNTIME, SLST,SYST, ZODI_FGR, ZODI_FGR_ PROD*	RFC1, SDIFRUNTIME, SLST,SYST, ZODI_FGR, ZODI_FGR_ PROD*
	RFC_TYPE	FUGR	FUGR	FUGR	FUGR
S_TCODE	TCD	SU53	SE38,SU53	SE38,SU53	SE38,SU53
S_BTCH_ADM	BTCADMIN	Y	Y	Y	Y
S_BTCH_JOB	JOBACTION	RELE	RELE	RELE	RELE
	JOBGROUP	'	'	'	'
S_CTS_ADMI	CTS_ADMFCT	TABL	TABL	TABL	TABL
S_DATASET	ACTVT	06,33,34	06,33,34	06,33,34	06,33,34
	FILENAME	\\10.30.0.20 1\ODI_ SAPTEST\ZODI *, \\HYPTTEST01\ sapmnt*	*\\10.30.0.2 01\ODI_ SAPTEST*, \\10.30.0.20 1\ODI_ SAPTEST\ZODI _233030*	*\\10.30.0.2 01\ODI_ SAPTEST*, \\10.30.0.20 1\ODI_ SAPTEST\ZODI _233030*	*\\DEL-7TMK2 BS\odiagentb ox*, \\DEL-7TMK2B S\odiagentbo x, \\DELR6\sap mnt*
	PROGRAM	SAPLSLOG, SAPLSTRF, SAPLZODI*, ZODI*	SAPLSLOG, SAPLSTRF, SAPLZODI, ZODI*	SAPLSLOG*, SAPLSTRF*, SAPLZODI*, ZODI*	SAPLSLOG, SAPLSTRF, SAPLZODI*, ZODI*
S_DEVELOP	ACTVT	01	01,02,03,06, 16	01,02,03,06, 16	01
	DEVCLASS				
	OBJNAME	ZODI_FGR_ PROD*	ZODI_FGR_ PROD*	ZODI_FGR_ PROD*	ZODI_FGR_ PROD*
	OBJTYPE	FUGR	FUGR	FUGR, PROG	FUGR
	P_GROUP				
S_PROGRAM	P_ACTION		EDIT, SUBMIT, VARIANT	EDIT, SUBMIT, VARIANT	
	P_GROUP				

Object	Field Name	Value (4.6c)	Value (4.7)	Value (ECC 5)	Value (ECC 6)
S_TRANSPORT	ACTVT	01,02,03	01,02,03	01,02	01,02
	TTYPE	DTRA,TASK	DTRA,TASK	DTRA,TASK	DTRA,TASK
S_APPL_LOG	ACTVT	06	06	06	06
	ALG_OBJECT	ZODI_APPOBJ	ZODI_APPOBJ	ZODI_APPOBJ	ZODI_APPOBJ
	ALG_SUBOBJ	ZODI_SUBOBJ	ZODI_SUBOBJ	ZODI_SUBOBJ	ZODI_SUBOBJ

B.1.7 Authorizations Required for LKM SAP ERP Execution as Background Process in Production

The following SAP authorizations are required for executing LKM SAP ERP as background process in a SAP development environment.

Object	Field Name	Value (4.6c)	Value (4.7)	Value (ECC 5)	Value (ECC 6)
S_RFC	ACTVT	16	16	16	16
	RFC_NAME	RFC1,SDIF,SY ST,SLST, ZODI_FGR_ PROD*	SDIF,SDIFRUN TIME,SLST,SY ST,ZODI_FGR_ PROD*	RFC1,SDIFRUN TIME,SLST, SYST,YSU, ZODI_FGR_ PROD*	RFC1,SDIFRUN TIME,SYST,ZO DI_FGR_PROD*
	RFC_TYPE	FUGR	FUGR	FUGR	FUGR
S_TCODE	TCD	SU53	SU53	SU53	SU53
S_BTCH_ADM	BTCH	Y	Y	Y	Y
S_BTCH_JOB	JOBACTION	RELE	RELE	RELE	RELE
	JOBGROUP	' '	' '	' '	' '
S_DATASET	ACTVT	06,33,34	06,33,34	06,33,34	06,33,34
	FILENAME	\\10.30.0.20 1\ODI_ SAPTEST\ZODI *, \\HYPTTEST01\ sapmnt*	\\10.30.0.20 1\ODI_ SAPTEST*, \\DEL-7TMK2B S\odiagentbo x*, \\DELR66\sap mnt*	\\10.30.0.20 1\ODI_ SAPTEST*, \\DEL-7TMK2B S\odiagentbo x*, \\DELR66\sap mnt*	\\10.30.0.20 1\ODI_ SAPTEST*, \\DEL-7TMK2B S\odiagentbo x*, \\DELR66\sap mnt*
	PROGRAM	SAPLSLOG,	SAPLSLOG,	SAPLSLOG,	SAPLSLOG,
		SAPLSTRF,	SAPLSTRF,	SAPLSTRF,	SAPLSTRF,
SAPLZODI*, ZODI*		SAPLZODI*, ZODI*	SAPLZODI*, ZODI*	SAPLZODI*, ZODI*	
S_APP_LOG	ACTVT	06	06	06	06
	ALG_SUBOBJ	ZODI_SUBOBJ	ZODI_SUBOBJ	ZODI_SUBOBJ	ZODI_SUBOBJ
	ALG_SUBOBJ	ZODI_SUBOBJ	ZODI_SUBOBJ	ZODI_SUBOBJ	ZODI_SUBOBJ

B.2 SAP Connection Test

This connection test should be performed after configuring the SAP ABAP data server in the topology.

This test performs the following operations:

- It establishes a test connection to the configured SAP system.
- It validates the proper setup of SAP JCo by displaying the About dialog.

This test is performed using an RKM called RKM SAP ERP Connection Test. This KM does not perform any reverse operation. It simply runs the connection test.

To run the connection test:

1. Create an SAP ERP Model based on the SAP ABAP technology and on the SAP ABAP logical schema using the standard procedure, as described in "Creating a Model" of the *Developing Integration Projects with Oracle Data Integrator*.
2. In the Reverse Engineer tab of the SAP ERP Model, select the RKM SAP ERP Connect Test.
3. Save the model.
4. Click **Reverse-Engineer** to start the reverse.

Within a few seconds, the SAP JCo About dialog should appear. If not, review the session execution log in the Operator Navigator. Please verify that the output of task "Test SAP Connection" contains Successfully connected to SAP System. If not, the connection test has failed and the connection problem must be analyzed.

B.3 SAP Stand-Alone Connection Test

In addition to the Connection Testing, a test can be performed outside of ODI using a standalone java utility. This utility is available with the Standalone agent.

To use a standalone java utility to test the connection:

1. Open a command window.
2. Go to the <ODI_HOME>/odi/sdk/lib directory.
3. Make sure that JAVA_HOME points to a supported JVM.
4. Make sure that you have installed SAP Java Connector and that the sapjco3.jar and the sapjco3 library are in the <ODI_HOME>/odi/sdk/lib directory.
5. Launch the utility using the following command:

On Windows:

```
java -cp sapjco3.jar;odi-sap.jar oracle.odi.sap.km.test.JCoTest
```

On Linux/UNIX:

```
java -cp sapjco3.jar:odi-sap.jar oracle.odi.sap.km.test.JCoTest
```

This command generates a ODI_SAP_CON_POOL.jcoDestination file in the folder <ODI_HOME>/odi/sdk/lib folder.

For more information, see "Appendix C, SAP Stand-Alone Connection Test" in the *Getting Started with SAP ABAP ERP Adapter for Oracle Data Integrator*.

6. Use a text editor to open the ODI_SAP_CON_POOL.jcoDestination file. This file should look as follows:

```
#for tests only!
jco.client.lang=EN
jco.destination.peak_limit=10
jco.client.client=800
jco.client.passwd=<SAP Password>
jco.client.user=<SAP User>
```

```
jco.client.sysnr=00
jco.destination.pool_capacity=5
jco.client.ashost=<SAP Application Server>
```

7. Enter you SAP connection information, which you have received from your SAP administrator.
8. Launch the utility using the same command. The utility uses the file that you have edited, and outputs the test results or the possible issues.

In addition to just testing the SAP connection, the utility will also validate the existence of certain Function Modules required for the RKM. These are installed during first execution of the RKM (UPLOAD_ABAP_BASE and UPLOAD_ABAP_CODE set to true).

9. Delete the ODI_SAP_CON_POOL.jcoDestination file after execution, as it contains the SAP login credentials.

Installing ODI SAP Components

This appendix describes how to update, install, and uninstall ODI SAP components. It also provides information on how to validate the ODI SAP setup.

This appendix includes the following sections:

- [Section C.1, "Updating ODI SAP Components"](#)
- [Section C.2, "Installing ODI SAP Components"](#)
- [Section C.3, "Validating the ODI SAP Setup"](#)
- [Section C.4, "Uninstalling ODI SAP Components"](#)

C.1 Updating ODI SAP Components

During first-time installation the RKM installs some ODI objects into the SAP system. This installation consists of two parts: some base objects and some RFCs. When the RKM options `UPLOAD_ABAP_BASE` and `UPLOAD_ABAP_CODE` are both set to `true`, the base objects and the RFCs are installed. Such full installation requires that no ODI SAP objects are installed in the SAP system.

See [Uninstalling ODI SAP Components](#) for information about how to uninstall ODI SAP Components if needed.

If the ODI objects installed into your SAP systems have been installed by RKM SAP ERP v32 or later, or RKM SAP BW v23 or later, there is no need for reinstalling the base objects and it is sufficient to update just the RFCs. Reinstalling the RFCs is achieved by executing a reverse engineering with the RKM option `UPLOAD_ABAP_BASE` set to `false` and RKM option `UPLOAD_ABAP_CODE` set to `true`.

C.2 Installing ODI SAP Components

Installation of the ODI SAP components is done through the use of TRs (SAP Transport Requests). These transport requests also contain the SAP user profiles. This section describes how to import the ODI SAP components and grant the required authorizations to the ODI SAP user by assigning the ODI SAP profiles.

This section includes the following topics:

- [Appendix C.2.1, "Installing SAP Transport Request \(TR\)"](#)
- [Appendix C.2.2, "Installing and Assigning SAP User Profile"](#)

C.2.1 Installing SAP Transport Request (TR)

Installation of SAP Transport Request requires downloading the TRs from Bristlecone e-delivery website.

This section includes the following topics:

- [Appendix C.2.1.1, "Downloading the Transport Request files"](#)
- [Appendix C.2.1.2, "Installing the Transport Request Files"](#)

C.2.1.1 Downloading the Transport Request files

Perform the following steps to download the TR files:

1. Go to the following Download URL:

http://www.bristleconelabs.com/edel/showdownload.html?product=odi_sap_km_transport

2. Select the required file and click **Download**. You will be redirected to Login Page.

If you are a registered user, log in with the registered Email Id as user name and the password received.

If you are a new user, perform the following steps to register:

- a. Go to the following URL:

<http://www.bristleconelabs.com/edel/Register.html>

- b. Click **Register**.

- c. Fill in the details in the registration form.

In the **Email ID** field, provide a valid business email address. This will be used to email your password.

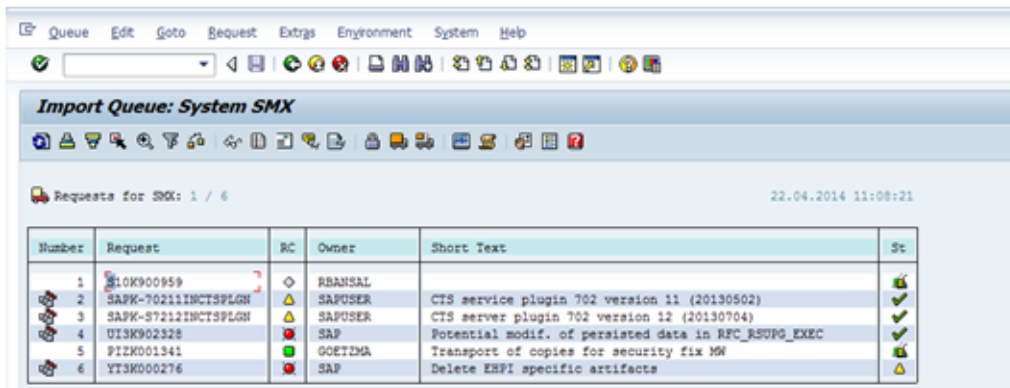
- d. Reopen the download link mentioned in step 1.

- e. Select the appropriate SAP version of the TR and download.

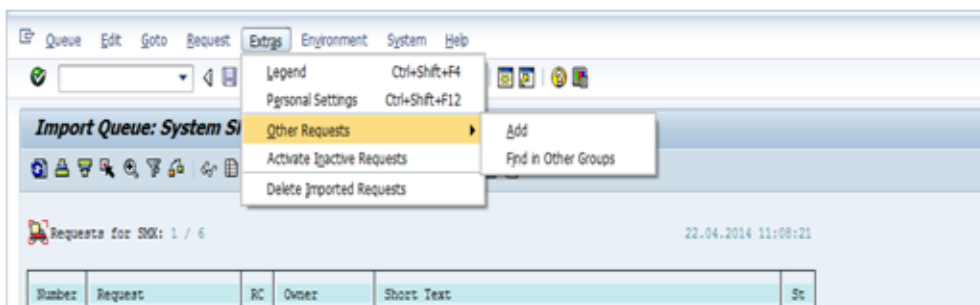
C.2.1.2 Installing the Transport Request Files

Perform the following steps to install the TR files:

1. Log in to the desired SAP System where you want to import the Transport Request.
2. In the command field, type Tcode `STMS` and press **Enter**.

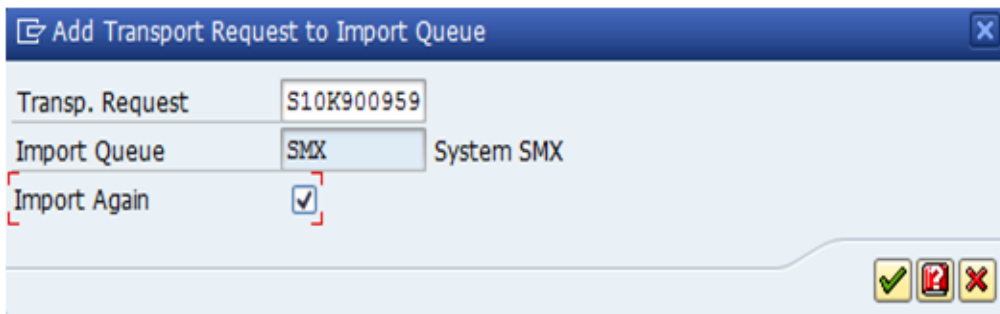



- If you want to add other Transport Request, click **Extras> Other Requests> Add** on the menu.

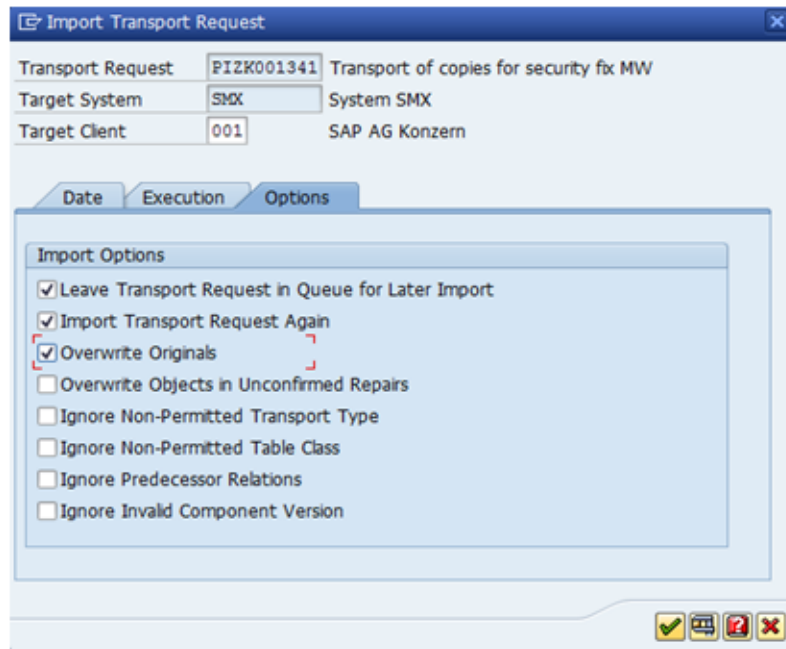


The added Transport Request can be seen in the System Import Queue.

- Type the Transport Request Number, click **Import Again** check box, and then click **Continue**.

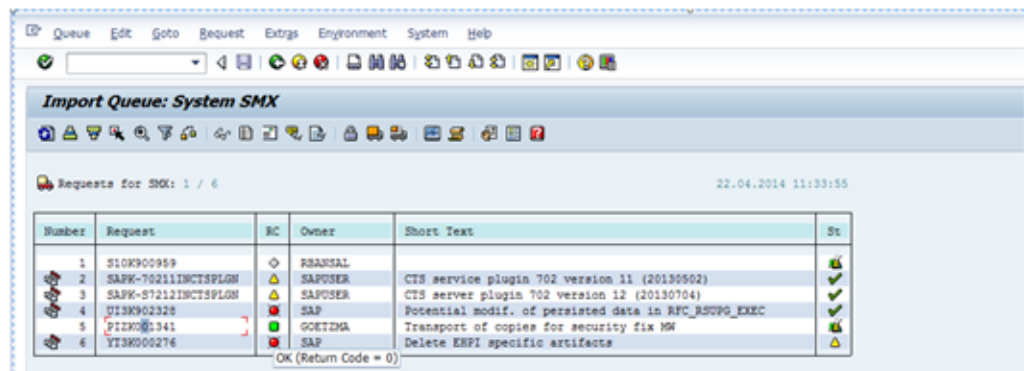


- Select the Transport Request and click the  icon or press **Ctrl + F11**.
- Specify your target client, for example, Client is 001.
- Click the **Options** tab, choose first three options and click **Continue**.



11. On the Start Import dialog, click **Yes**.

The Transport Request import will start. Wait until the Transport Request import is completed successfully as shown below.



Caution: Artefacts of TR Files: Naming convention used is R<6 digits>.<source system> and K<6 digits>.<source system>.

The Source system is the developers source system.

K Type Transport : Cofile (1 - 3 KB in size)

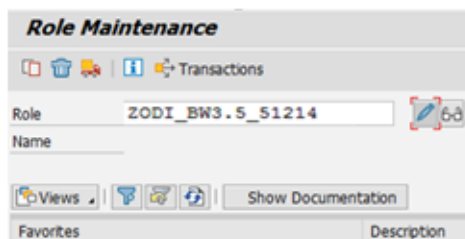
R Type Transport : Datafile (size more than the Cofile)

- **K file is a Cofile:** All transport requests' control and metadata attributes gets stored in this file, which are in the format of R<T.RNo>.<SID>. These are command or change request information files that include information about the transport type, object classes, required import steps, and post-processing exit codes.
 - **R file is a Datafile:** All transport requests' data gets stored in this file, which are in the format of K<T.RNo>.<SID>. This contains the actual data for the Transport i.e. what changes will be made in your system.
-

C.2.2 Installing and Assigning SAP User Profile

Perform the following steps to install and assign SAP user profile:

1. Execute PFCG, on the menu, click **Role** and then select **Upload**.
2. Click **Continue** on the Information dialog.
3. Specify the path where you have saved the .SAP file and then click **Open**.
4. Scroll down the drop down list and provide the appropriate input and click **Allow**.
5. Click **Continue** on the subsequent screen.
Role is successfully uploaded.
6. Type the name of uploaded role and click the **Edit** icon as shown below.



7. Click the **Roles** tab.
8. Perform the following steps to activate each role:
 - a. Double click the role Z:ODI_ANU_LBU.
 - b. Click the **Authorizations** tab.
 - c. Click the **Change Authorization Data** icon shown below.

- d. Click the **Generate**  icon.

- e. Click **Generate** and then click **Back**.
 f. Click the **User** tab and compare the user.

Note: The status of Authorizations and User tab must be green.

- g. Repeat these steps to activate for all roles inside Z:ODI_COMPOSITE.
 9. Click **Save**.

10. Once the import process is done successfully, assign the imported profile to the SAP ODI user.
11. Request and assign SAP Developer License Key to ODI SAP User as described in section [Section A.1, "SAP ABAP BW Required Privileges"](#) or [Section B.1, "SAP ABAP ERP Required Privileges"](#). This is mandatory for the ODI SAP Adapter to work.

Note: In a few use-cases (use in production) the developer key can be omitted. For details refer to [Section A.1, "SAP ABAP BW Required Privileges"](#) or [Section B.1, "SAP ABAP ERP Required Privileges"](#).

C.3 Validating the ODI SAP Setup

This section describes some basic validation steps for the SAP administrators. Executing these steps ensures that previous installation steps were successful.

This section includes the following topics:

- [Appendix C.3.1, "Validating the Shared Folder Setup"](#)
- [Appendix C.3.2, "Validating the FTP Setup"](#)
- [Appendix C.3.3, "Validating SAP Privileges"](#)
- [Appendix C.3.4, "Validating SAP Transport Layer Name"](#)

C.3.1 Validating the Shared Folder Setup

Validating a shared folder setup applies only if you plan to transfer data using a Shared Directory. This section can be skipped, if the FTP transfer is used. The validation of the shared folder setup needs to be performed before any subsequent steps in this guide can be performed. This validation is typically performed by your SAP Basis team.

Step 1: Validating folder access from the SAP application server

1. Start the SAPGUI.
2. Use the ODI SAP user and password to connect to the SAP system and client.
3. Go to transaction AL11.
4. Select the shared folder.
5. Double-click the shared folder to test the directory declaration.

This should report a successful connection. If not, please contact your SAP basis team. Do not continue until this test passes.

Note: SAP uses the OS user `<sid>adm` to connect to the directories declared in AL11.

6. Print screen.

Step 2: Validating folder access from the ODI agent machine

1. Log in to the ODI agent machine using the Windows user ID used for executing the ODI agent.

2. Open shared directory path in Windows Explorer.

This should list the content of the shared folder. Please make sure that you do NOT need to enter any credentials. If this does not work or you had to enter credentials, please contact your windows administrator or the system administrator of the system your shared folder is physically located on. Do not continue until this test passes.

3. Print screen.

C.3.2 Validating the FTP Setup

Validating an FTP setup applies only, if you plan to transfer data using FTP. This section can be skipped, if you use a Shared Directory for the data transfer. The validation of the FTP setup needs to be performed before any subsequent steps in this guide can be performed. This validation is typically performed by your SAP Basis team.

Step 1: Validating SAPFTP destination

1. Start the SAPGUI.
2. Use the ODI SAP user and password to connect to the SAP system and client.
3. Go to transaction SM59.
4. Expand TCP/IP connections.
5. Open the SAPFTP destination.
6. Click **Test connection**.

This should report a successful connection. If not, please contact your SAP basis team. Do not continue until this test passes.

7. Print screen.

Step 2: Testing FTP connection

1. Go to transaction SE38.
2. View the function module RSFTP002.
3. Hit F8 to run the ABAP program.
4. Enter the FTP userID and password.
5. Enter the FTP server host name or IP address.
6. Enter cd / or cd <ODI target directory>.
7. In the RFC_DESTINATION field, enter SAPFTP.
8. Hit F8 to run the test.

This should report a successful connection. The message should be similar to the following:

```
250 CWD successful.
```

If this test is not successful, please contact your SAP basis team. Do not continue until this test passes.

9. Print screen.

C.3.3 Validating SAP Privileges

This section describes how to test some of the key SAP privileges. Proceed with the subsequent steps in this guide only after successful validation of these tests. This validation is typically performed by your SAP Basis team.

Perform the following steps to validate whether a SAP user has appropriate dev rights and owns a dev license key:

1. Start SAPGUI.
2. Use the ODI SAP user and password to connect to the SAP system and client.
3. Go to transaction SE38.
4. Enter any sample program name like ZSAP_TEST in the program name field.
5. Click **Create**.
6. Perform similar tests for the transaction SE37 and SE11.

If a transaction allows the creation of a program without asking for any key or other authorization message, then the SAP user has validated that it has the appropriate dev rights and license key. Otherwise your SAP basis team needs to register the SAP user in service.sap.com to get the license key and a Basis person can help him with dev rights.

C.3.4 Validating SAP Transport Layer Name

As the SAP connector creates SAP objects, such as, for example, function modules, into the SAP development system, these changes need to be transported into QA and production systems once the development is done. The SAP's change and transport system uses the *SAP Transport Layer Name* to identify the route a change has to take. A transport layer is assigned to each development class and thus to all objects in that class. The transport layer determines:

- In which SAP System developments or changes to the repository objects are made
- If objects are transported to other systems within the group when development work has been completed

A consolidation route is created from the development system to the quality assurance system through the `transport layer Z<SID>`. It then becomes the standard transport layer for customer development and customizing.

A consolidation route is created from the development system to the quality assurance system through the `transport layer SAP` for the transport of SAP Standard objects.

It is important to specify the correct transport layer name before running the RKM SAP ERP for the first time.

Perform the following steps to identify the list of defined transport layers in your SAP landscape:

1. Log on in client 000 in the SAP System serving as the transport domain controller via transaction STMS.
2. Select **Overview > Transport Routes**. The Display Transport Routes dialog is displayed.
3. Select **Goto > Graphical Editor**.
4. To switch the mode, select **Configuration > Display <-> Change**.
5. Position the cursor on the SAP System.

6. Select **Edit > System > Change**. The Change System Attributes dialog is displayed.
7. Select the StandardTransport Layer tab.
8. Change the transport layer of the SAP System.
9. The result is the list of the different transport Layers.

By default, the RKM option `SAP_TRANSPORT_LAYER_NAME` is set to `SAP`. Ask your SAP basis admin which transport layer you should use. This transport layer name must be set on the `SAP_TRANSPORT_LAYER_NAME` RKM option. A wrong or invalid transport layer name will cause serious delays during the installation process.

C.4 Uninstalling ODI SAP Components

During first-time installation, the RKM installs some ODI objects into the SAP system. This installation consists of two parts, Base objects and RFCs. When the RKM options `UPLOAD_ABAP_BASE` and `UPLOAD_ABAP_CODE` are both set to `true`, the base objects and the RFCs are installed. Such full installation requires that none of the ODI SAP objects should already be installed in the SAP system. You will need to un-install ODI SAP Components.

Perform the following steps to uninstall ODI SAP components:

1. Start the SAP GUI.
2. Connect to the SAP systems you want to uninstall.
3. Go to the transaction `SE80` and provide Package name, for example, `ZODI_LKM_PCKG`.
4. Right click on package name, and select **other options > rebuild object list** from the context menu.
5. Expand the object list and delete individual objects in the order listed below:
 - Program
 - Function group
 - Table type
 - Structure
 - Message class

If asked for transport request, create a new request and copy the requested number in Notepad.

6. Repeat step 4.

If any object is remaining, it will be displayed. Delete the object.
7. Right click the package name, go to **display object directory entry > press lock overview** button, copy the request number, and save it in Notepad.
8. Go to transaction `SE10`, and release all the transport requests.
9. Go to Transaction `SE16`, enter package name in **DEVCLASS** field of table `TADIR`.

There should be a single record for the package name. If there are entries in the table, then go to `SE11` and table `E071`. Specify the Object and object name whose entry is in the `TADIR`.
10. Mark the field **LOCKFLAG** as **X** and execute.

From there you will get the task or request. Go to Tcode SE01 and release this request.

11. Go to transaction SLG0, and search for zodi* select message class and go to sub-objects and delete the object.

Now press back button and delete message class. Press **Save** button. It will ask for transport request, create a new request and save it in Notepad.

12. Delete the package from SE80.

It will ask for a transport request. Create the request and release it using transaction SE10.

13. Repeat the steps 3 to 12 for package ZODI_RKM_PACKAGE.

If any of the objects is not deleted, then go to SU53 and check missing privileges. Try to grant the same to that SAP user and then continue with deleting that object.

See [Appendix C.4.1, "Validating Uninstallation of ODI SAP Components"](#) for information on how to validate uninstallation of ODI SAP components.

C.4.1 Validating Uninstallation of ODI SAP Components

Perform the following steps to validate the uninstallation of the ODI SAP components:

1. Go to transaction SE11.
2. Enter the table name TADIR and click **Display**.
3. On the Application toolbar, click **Contents**.
4. Specify the package name in the field **DEVCLASS** and press **F8**.
If there are no entries, a message **no table entries found** is shown.
If any entries are listed, the uninstallation is incomplete and reinstallation must not be attempted.
5. Go to transaction SE11. Enter the table name T100 and click **Display**.
6. On the Application toolbar, click **Contents**.
7. Type zodi* in the field **ARBGB** and press **F8**.
If there are no entries, a message **no table entries found** is shown.
If any entries are listed, the uninstallation is incomplete and reinstallation must not be attempted.

Moving ODI and SAP Components from Development to Production

This appendix describes how to move ODI and SAP Components from the Development environment to the Production environment. In this appendix, system ECC 5.0 is used.

This appendix includes the following sections:

- Section D.1, "Transport Request (TR)"
- Section D.2, "Remote Function Call (RFC)"
- Section D.3, "Adding Mappings Under Same Transport Request"
- Section D.4, "Creating ODI Package"
- Section D.5, "Generating ODI Scenario"
- Section D.6, "Releasing Transport Request to Production"
- Section D.7, "Releasing Transport Request of the Mapping to Production"
- Section D.8, "Executing the ODI Scenario"

D.1 Transport Request (TR)

The ODI SAP Adapter uses different sets of SAP objects, which are bundled in TRs:

- **ODI SAP Adapter product TRs:** There are two product TRs, which are available for download and are installed during the ODI SAP Adapter product installation:
 - TR containing SAP objects relevant for **development**, for example, to allow ODI to retrieve metadata from SAP system.
 - TR containing SAP objects relevant for **production**, for example, to run and monitor data extraction jobs.
- **ODI generated TRs:** During development the ODI SAP Adapter generates ABAP programs. These programs can be (automatically) uploaded and will be part of TRs. Depending on users choice there can be one or several TRs containing ABAP programs for data extraction.

This sections describes these TRs and how they can be inspected in SAP.

Note: Please note the following:

- TR is released by SAP BASIS team.
- There is a possibility that TR may automatically move to Production. This happens if Transport Route is configured. In most cases, the Transport Route to Production from Development or QA is not configured, since manually importing the TR is considered better. In case of an automatically configured TR, emphasis should be on correct sequence of TR, since the sequence plays an important role.
- In a distributed Production environment, the workbench TR should be imported in all of the application servers.

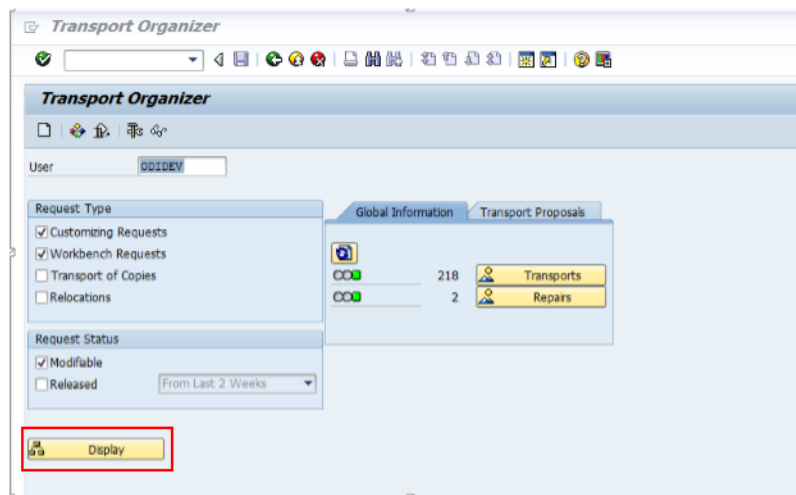
This section includes the following topics:

- [Appendix D.1.1, "Viewing the List of TRs Created after RKM Execution"](#)
- [Appendix D.1.2, "Transport Request \(TR\) relevant to DEV"](#)
- [Appendix D.1.3, "Transport Request \(TR\) relevant to PROD"](#)

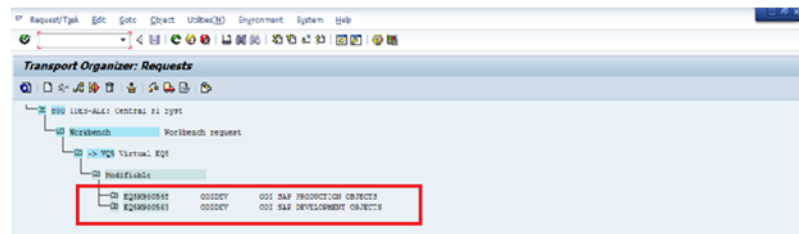
D.1.1 Viewing the List of TRs Created after RKM Execution

Perform the following steps to view the list of TRs that are created after RKM execution:

1. Type TCODE SE10.
2. Click the **Display** button that is located towards the bottom left corner of the screen.



Under Workbench, different target systems are listed. For example, VQ5, as shown below.



3. Expand the target system (example, VQ5), associated with the transport layer name (example, SAP) that was used at the time of RKM installation. The following two Transport Requests as shown above are displayed:
 - ODI SAP PRODUCTION OBJECTS
 - ODI SAP DEVELOPMENT OBJECTS

See [Appendix D.1.1.1, "Description of TR Fields"](#) for the description of TR fields.

D.1.1.1 Description of TR Fields

The following list describes the different TR fields.

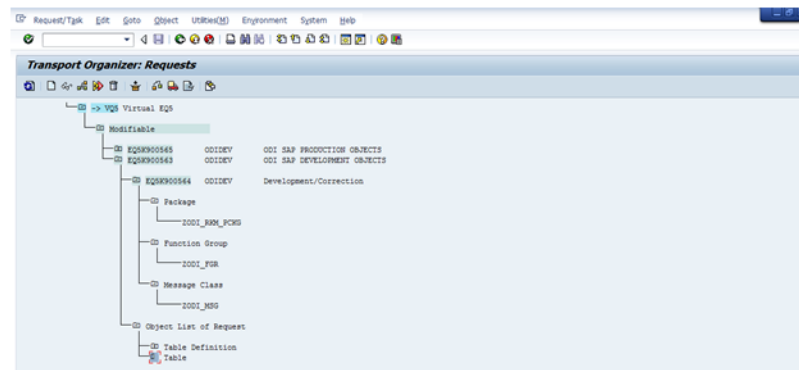
- **TR Name:** EQ5K900562

Note: EQ5K900562, where EQ5 is the system identification (SID), K is a keyword, and the number is automatically generated from a range by the system, which starts at 900001 and is not to be maintained by the system administrators.

- **TR Description:** ODI SAP PRODUCTION OBJECTS
- **Transport Layer Name:** SAP

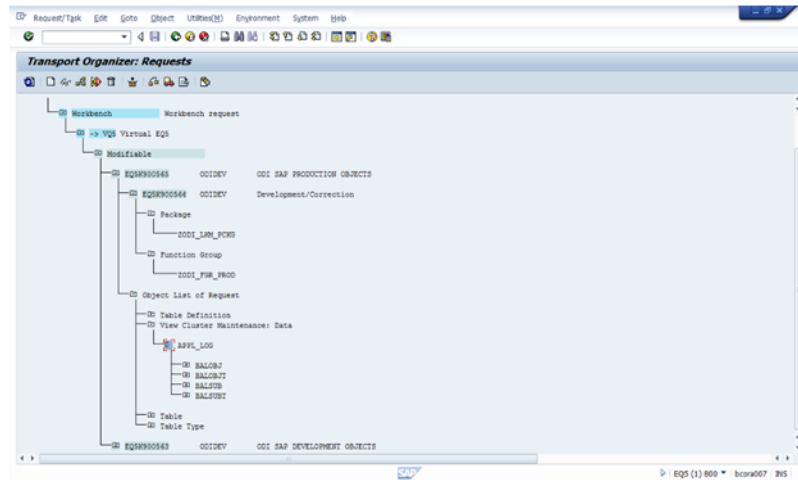
D.1.2 Transport Request (TR) relevant to DEV

The TR with description ODI SAP DEVELOPMENT OBJECTS is relevant to Development. Expanding this will display **Package**, **Function Group**, and **Message Class** that are created under this TR.



D.1.3 Transport Request (TR) relevant to PROD

The TR with description ODI SAP PRODUCTION OBJECTS is relevant to Production. Expanding this will display **Package**, **Function Group**, **Object List of Request**, for example, APPL_LOG that are created under this TR.



D.2 Remote Function Call (RFC)

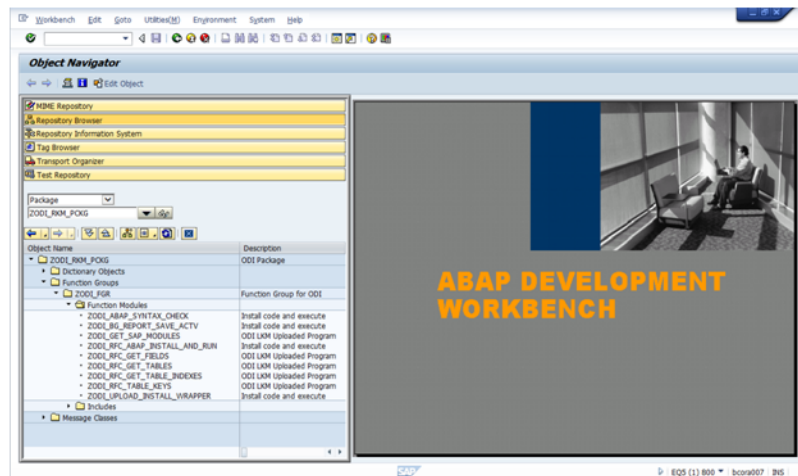
The entire set of generated RFCs are grouped under two SAP packages, based upon their execution environment. The following two SAP packages are created:

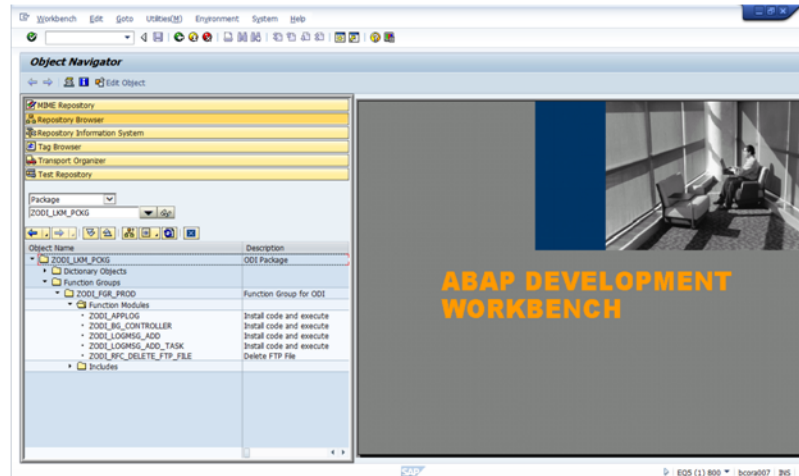
- **ZODI_RKM_PCKG**: contains all the RFCs required by the development SAP systems.
- **ZODI_LKM_PCKG**: contains all the RFCs required only by the production SAP system.

The list of RFCs created into SAP system can be seen by using TCODE SE80.

Perform the following steps to see a list of RFCs created in the SAP system:

1. Open TCODE SE80.
2. Below the option **Test Repository**, from the first drop-down list, select **Package**.
3. Type the package name (ZODI_RKM_PCKG/ZODI_LKM_PCKG) and press **Enter**.
4. Under **Object Name** segment, expand **Function Groups >ZODI_FGR >Function Modules**. A list of installed RFCs is displayed.





D.3 Adding Mappings Under Same Transport Request

By default the ODI SAP Adapter creates one TR per mapping. In case multiple mappings are used in a scenario, the user may want to move the generated ABAP programs inside SAP as a single unit. This is best achieved by grouping all ABAP programs into a single TR. This section describes how to let ODI group multiple ABAP programs into the same TR.

For background processing ODI combines all ABAP programs having the same `SAP_REPORT_NAME_PREFIX` into the same TR. The very first mapping executed defines the TR description in SAP via the KM option `SAP_TRANSPORT_REQUEST_DESC`. See [Section D.3.1, "KM Options Descriptions"](#) for more details.

Perform the following steps to add interfaces under the same Transport Request:

1. Create two mappings. For example, EXTRACT MAKT OBJECT DATA and EXTRACT MARA OBJECT DATA.
2. In the Flow tab, set the value of option `BACKGROUND_PROCESSING` to `TRUE`.

Options:

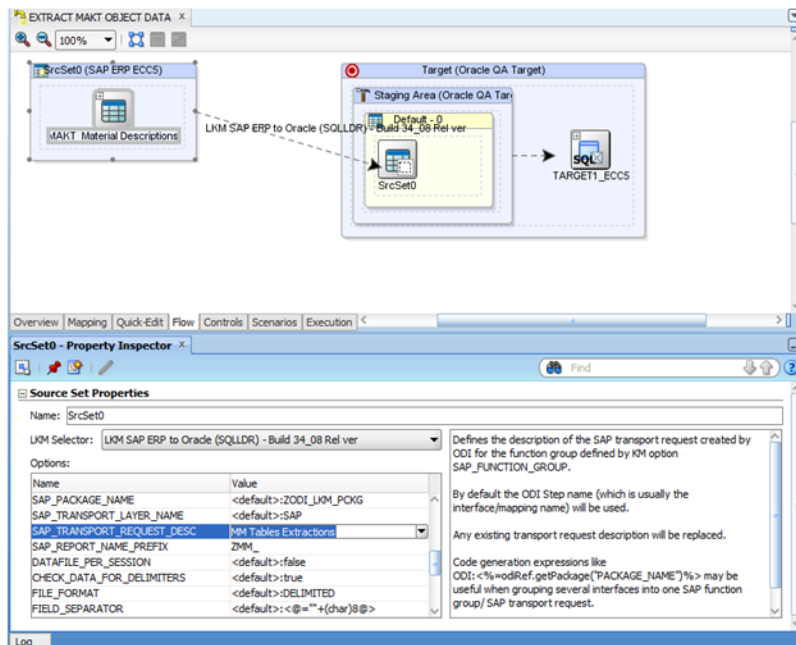
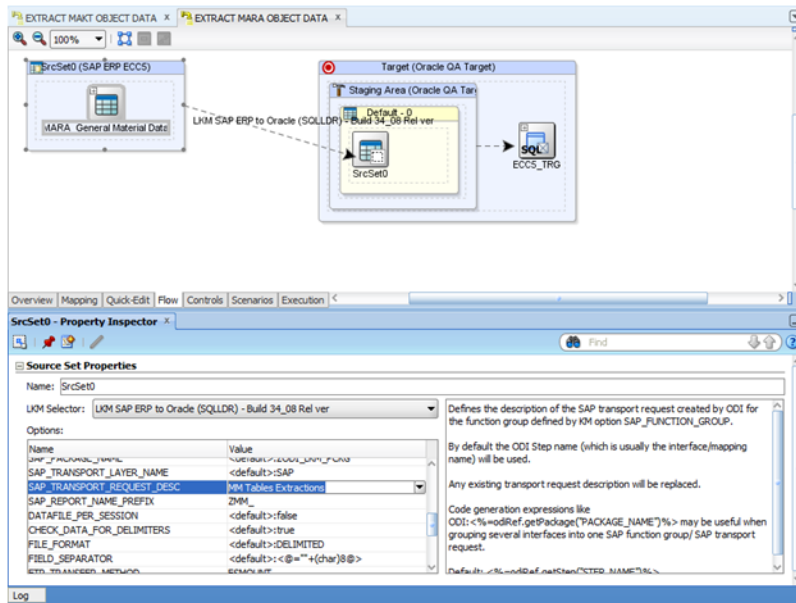
Name	Value
EXECUTE_ABAP_CODE	<default>:true
ABAP_PROGRAM_NAME	<default>:
BACKGROUND_PROCESSING	<default>:true
JOB_CLASS	<default>:A
LOG_FILE_NAME	<default>:<?=new File((sBIAppsTempPath.equ...
TEMP_DIR	<default>:<?=new File((sBIAppsTempPath.equ...
MAX_ALLOWED_ERRORS	<default>:1
DELETE_TEMPORARY_OBJECTS	<default>:true

3. Enter the same `SAP_TRANSPORT_REQUEST_DESC`, for example, MM Tables Extractions, in both the mappings.

For description of `SAP_TRANSPORT_REQUEST_DESC` option, see [Appendix D.3.1, "KM Options Descriptions"](#).

4. Enter `SAP_REPORT_NAME_PREFIX`, for example, ZMM_, in both the mappings.

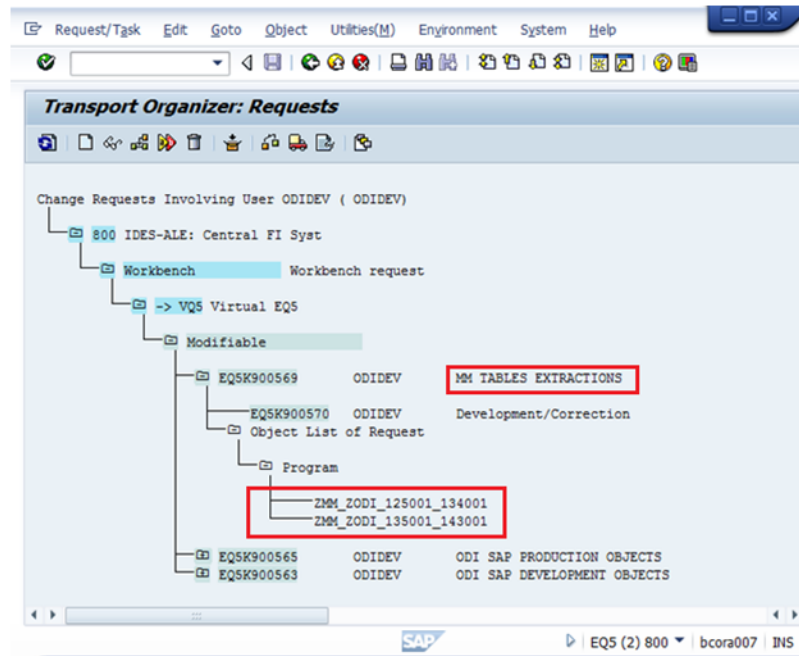
For description of `SAP_REPORT_NAME_PREFIX` option, see [Appendix D.3.1, "KM Options Descriptions"](#).



5. Execute both the mappings.
6. Go to SAP TCODE SE10.
7. Click the **Display** button.

The newly created transport request with the description provided in the KM option is displayed.

8. Expand the newly created **TR > Object List of Request > Program**, under this, the reports created by execution of LKM can be seen.



After following the above steps, both the mappings, ZMM_ZODI_125001_134001 and ZMM_ZODI_135001_143001 are added to the same transport request as shown in the image above.

Note: In the Mapping ZMM_ZODI_135001_143001,
ZMM_: is the SAP_REPORT_NAME_PREFIX.
ZODI_135001_143001: dynamic ABAP program name.

The grouping of the above two ABAP programs is because of same value for SAP_REPORT_NAME_PREFIX KM option.

D.3.1 KM Options Descriptions

This section provides the description of the KM options SAP_REPORT_NAME_PREFIX and SAP_TRANSPORT_REQUEST_DESCRIPTION.

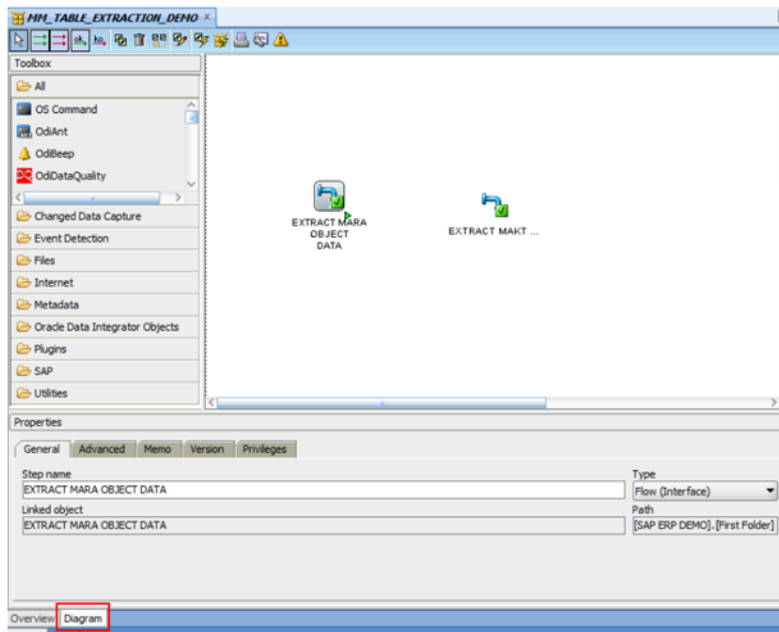
- **SAP_REPORT_NAME_PREFIX:** defines the prefix of all reports to be grouped into a common transport request, and is used in conjunction with ABAP_PROGRAM_NAME. Only applies if BACKGROUND=1. The prefix must start with Z. Please contact your SAP administrator for recommended report name prefixes.
- **SAP_TRANSPORT_REQUEST_DESCRIPTION:** defines the description of the SAP transport request created by ODI for the function group defined by KM option SAP_FUNCTION_GROUP. By default, the ODI Step name (which is usually the mapping name) will be used. Any existing transport request description will be replaced. Code generation expressions like ODI:<%=odiRef.getPackage("PACKAGE_NAME")%> may be useful when grouping several interfaces into one SAP function group/ SAP transport request.

Default: <%=odiRef.getStep("STEP_NAME")%>

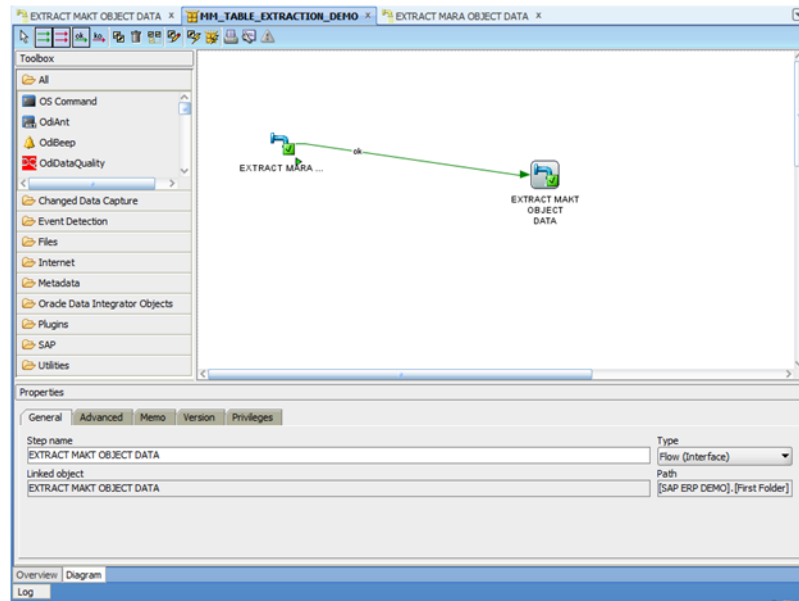
D.4 Creating ODI Package

Perform the following steps to create an ODI Package:

1. Go to SAP ERP project, for example, SAP ERP Demo, expand the **First Folder**, right-click on **Packages** and select **New Package**.
2. In the Definition window, enter the name of the package, for example, MM_TABLE_EXTRACTION_DEMO, and click **Save**.
3. Click the **Diagram** tab.
4. To define the steps in the package, select EXTRACT MARA OBJECT DATA and EXTRACT MAKT OBJECT DATA interfaces from **Designer window->Project->First Folder -> Interfaces**, and drag-and-drop into the **Diagram** tab. These components appear as steps in the package and are not sequenced yet.
5. To specify the first step in the package, select and right-click this step, then select **First Step** from the context menu. A small green arrow appears on this step.



6. Select the **Next Step on Success** icon from the diagram tool bar.
7. To sequence the steps with this tool, click on one step, then click on the next step to be executed.



D.5 Generating ODI Scenario

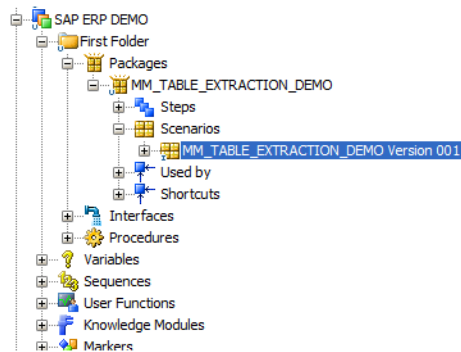
Perform the following steps to generate an ODI Scenario:

1. Select the MM_TABLE_EXTRACTION_DEMO Package.
2. Right-click and select **Generate Scenario**.

The New Scenario dialog box appears.

3. Click **OK**. Oracle Data Integrator processes and generates the scenario.

The generated scenario is listed under Scenarios as shown below.



D.6 Releasing Transport Request to Production

To release the transport request to production, the transport's request number associated with that production (ZODI_LKM_PCKG) package is required.

Perform the following steps to release a transport request to production:

1. Go to TCODE SE11 of SAP system, and enter E071 in **Database table** field.

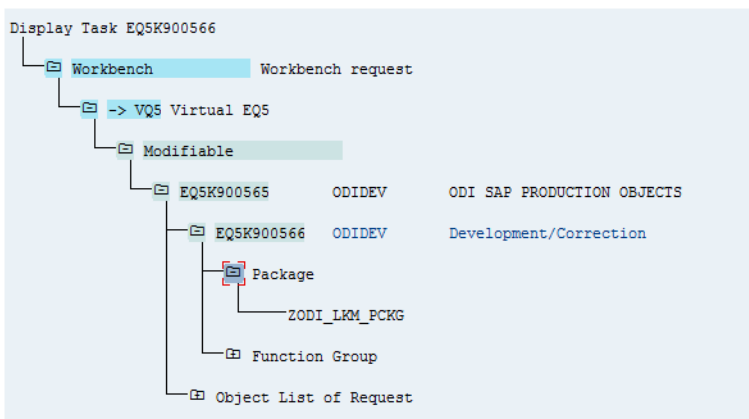
2. Click **Display**.
3. Click the **Content** icon.
4. Enter the package name ZODI_LKM_PCKG for OBJ_NAME and X for LOCKFLAG fields.
5. Press F8 or execute.
This gives the transport request number.

Table: E071
 Displayed Fields: 10 of 10 Fixed Columns: List Width 0250

TRKORR	AS4POS	PGMID	OBJECT	OBJ_NAME
EQ5K900566	000001	R3TR	DEVC	ZODI_LKM_PCKG

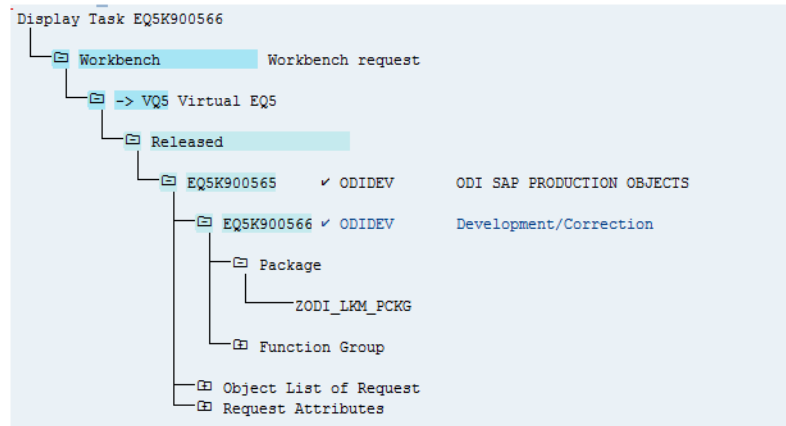
6. Select and Copy the transport request number.
7. Go to TCODE SE01 and enter the copied transport request number.

8. Click **Display**.
The request number and task number along with the objects under that request are shown.



9. Click on the task number below the request number and click the icon.
10. Click on the request number and click the icon.

Note: A check mark indicates that the transport request has been released.

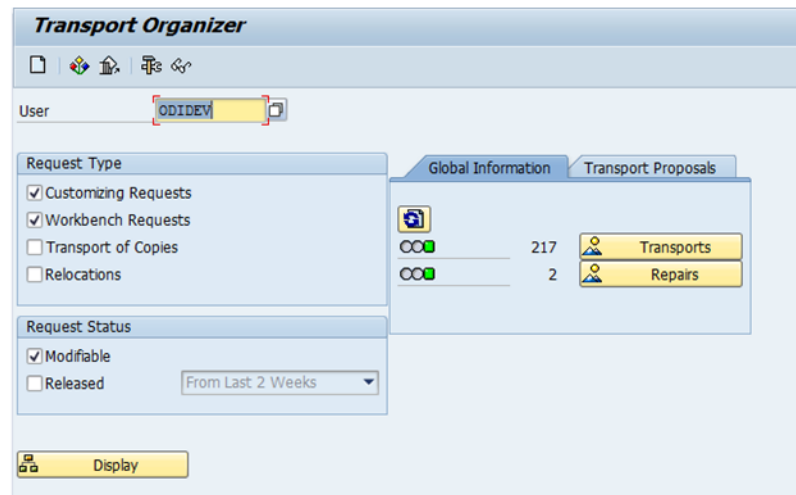


D.7 Releasing Transport Request of the Mapping to Production

To release the transport request of the mapping, the transport's request number associated with that mapping is required.

Perform the following steps to find the TR associated with the mapping:

1. Go to TCODE SE10 in SAP system, and enter the user name used for executing the LKM.
2. Change the **Request Status** to **Modifiable** by checking it.

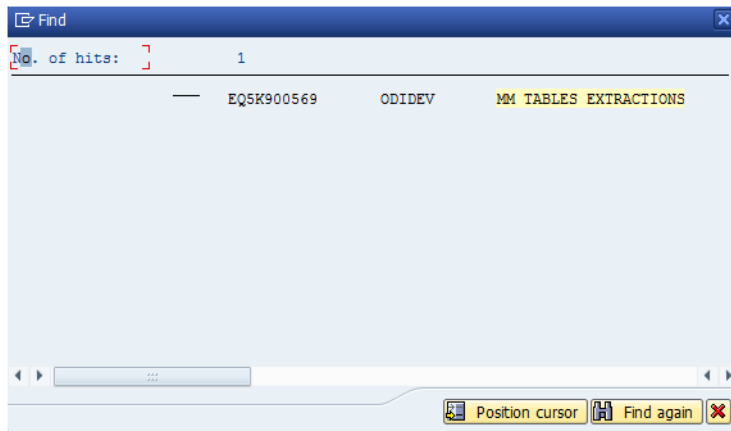


3. Click **Display**.
4. Press **CTRL+F** to search. Provide the SAP_TRANSPORT_REQUEST_DESC value which was entered in the interfaces while executing the LKM, for example, MM TABLES EXTRACTION.



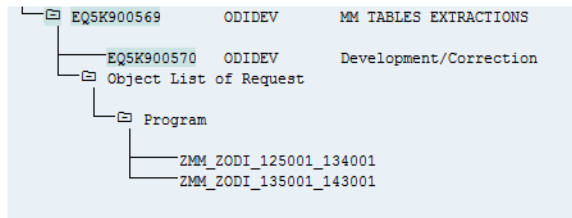
5. Click the Search icon.

A pop-up displaying the transport request number with its description appears.



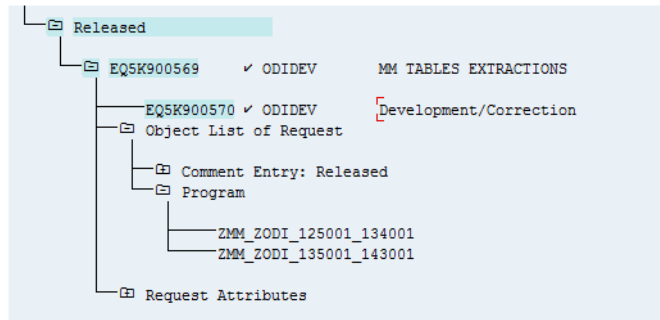
6. Double click on the Transport Request number.

The request number and task number along with the objects under that request are shown.



7. Click on the task number below the request number, and click the icon.
8. Click on the request number, and click the icon.

Note: A check mark indicates that the Transport Request has been released.

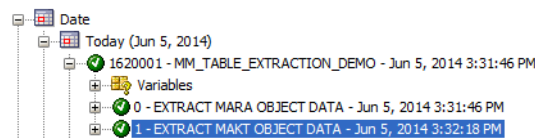


D.8 Executing the ODI Scenario

Perform the following steps to execute the ODI Scenario:

1. Right-click on the scenario MM_TABLE_EXTRACTION_DEMO Version 001.
2. Select **Execute** from the context menu.
3. Click **OK** on the **Execution** and **Information** dialogs.

The scenario execution report can be reviewed in Operator, and the same results are seen as those obtained when the package was executed.



It is also possible to review the scenario execution report from the Designer module, by expanding the scenario tree view.

