

信息技术最佳实践
ORACLE 核心应用技术
FUSION MIDDLEWARE
ODI应用指南

Author: 黄建华
Creation Date: March 7, 2009
Last Updated: January 5, 2010
Document Ref: <Document Reference Number>
Version: DRAFT 1A

Approvals:

<Approver 1>

<Approver 2>



Copy Number _____

Document Control

Change Record

Date	Author	Version	Change Reference
19-May-07	Jianhua.Huang	Draft 1a	No Previous Document

Reviewers

Name	Position

Distribution

Copy No.	Name	Location
1	Library Master	Project Library
2		Project Manager
3		
4		

Note To Holders:

If you receive an electronic copy of this document and print it out, please write your name on the equivalent of the cover page, for document control purposes.

If you receive a hard copy of this document, please write your name on the front cover, for document control purposes.

Contents

Document Control	ii
1. ODI基础	2
1.1. 参考资料	2
1.2. ODI基础	2
1.3. ODI理解之1	6
1.4. ODI组件	6
1.5. ODI安装	8
1.6. 服务和菜单	10
1.7. 基本训练	10
2. 完整的简单例子（资料库、体系结构、项目、模型、接口、包、方案）	11
2.1. ODI理解之2	11
2.2. 环境准备	12
2.3. 创建资料库	13
2.4. 创建物理体系结构	17
2.5. 创建逻辑体系结构	20
2.6. 创建项目	20
2.7. 创建模型	21
2.8. 创建接口	23
2.9. 运行接口	25
2.10. 监控和查看会话状态	26
2.11. 创建并运行包	27
2.12. 创建并运行方案	28
2.13. ODI理解之3	28
3. 最常用特性和功能实例一（CDC、Agent、Schedule）	30
3.1. ODI理解之4	30
3.2. ODI理解之5（CDC）	30
3.3. O2O CDC（Simple）	32
3.4. O2O CDC（Consistent Set）	35
3.5. O2O CDC（Consistent Set Using Log Minner）	41
3.6. Agent	42
3.7. Schedule	43
4. 最常用特性和功能实例二（SQL Server、XML）	45
4.1. Oracle to SQL Server	45
4.2. SQL Server to Oracle	46
4.3. XML to Oracle	47
5. Sequence、Variable、User Function、Procedure	50

5.1.	作用域.....	50
5.2.	Variable变量.....	50
5.3.	Sequence序列.....	51
5.4.	User Function自动义函数.....	53
5.5.	Procedure过程.....	54
5.6.	Procedure最简单的例子.....	55
5.7.	在Procedure中用序列、函数.....	56
6.	Knowledge Module.....	58
6.1.	客户化KM最佳简单例子.....	58
7.	Package及ODI工具箱.....	59
7.1.	概述.....	59
8.	Web Service.....	61
8.1.	安装Public Web Services.....	61
8.2.	设置Data Services.....	61
9.	FAQ&How To.....	62
9.1.	常见问题.....	62
9.2.	中英文名词.....	62
9.3.	对象加密.....	63
9.4.	Agent负载均衡.....	63
9.5.	Jython.....	63
9.6.	Substitution Methods.....	64
9.7.	常用代码块.....	64
9.8.	命令行工具.....	64
9.9.	升级.....	65
9.10.	用户权限.....	65
9.11.	安装Metadata Navigator.....	66
9.12.	安装Lightweight Designer.....	68
10.	专题.....	70
10.1.	DBLink.....	70
10.2.	对象冲突.....	70
11.	Open and Closed Issues for this Deliverable.....	71
	Open Issues.....	71
	Closed Issues.....	71

1. ODI基础

1.1. 参考资料

官方中文资料: <http://www.oracle.com/technology/global/cn/products/oracle-data-integrator/index.html>

官方英文资料: 安装盘下的index.htm, 或者OTN。

比较有用的文档, 收录在“Oracle Data Integrator应用指南.Source.rar”。

1.2. ODI基础

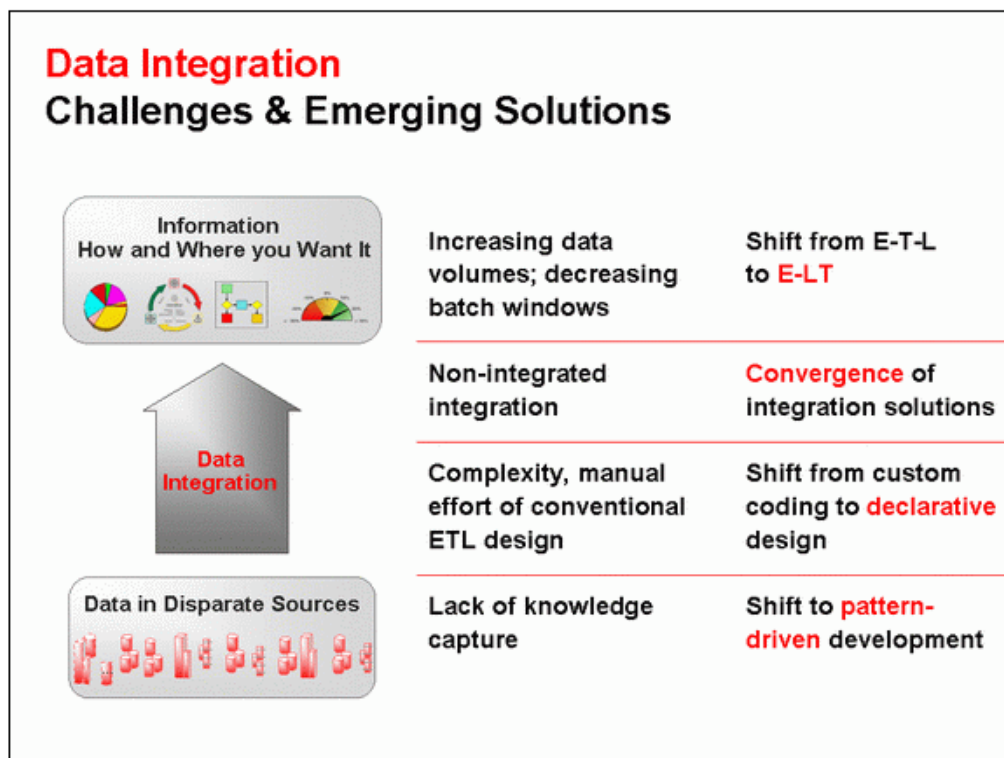
1.2.1. 简介

OD是Oracle在2006年10月收购Sunopsis公司后, 整合Sunopsis Active Integration Platform而推出的一款数据集成工具, 现在是Oracle Fusion Middleware的组件。

与OWB一样, ODI也是“ELT”而非“ETL”工具, Oracle不采用独立的引擎而是充分利用RDBMS的能力进行数据转换, 减少网络流量、平衡和提高性能的同时降低投入总成本。

1.2.2. 数据集成面临的挑战

数据集成面临的挑战和Oracle的解决方案:



1.2.3. Oracle的解决方案有何优势

Oracle的解决方案有何优势：

Oracle Data Integrator

Data Movement and Transformation from Multiple Sources to Heterogeneous Targets

Benefits

Performance

Flexibility

Productivity

Hot-Pluggability

Key Differentiated Features

Heterogeneous “E-LT”

Active Integration Platform

Declarative Design

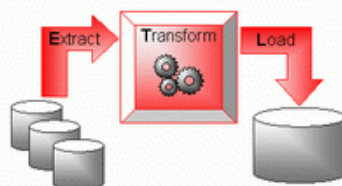
Knowledge Modules

1.2.4. ODI优势详解

1、ELT与传统的ETL相比，性能和成本都有很大优势

E-LT Architecture High Performance

Conventional ETL Architecture



Transform in Separate ETL Server

- Proprietary Engine
- Poor Performance
- High Costs
- IBM & Informatica's approach

Next Generation Architecture

“E-LT”



Transform in Existing RDBMS

- Leverage Resources
- Efficient
- High Performance

Benefits

- Optimal Performance & Scalability
- Easier to Manage & Lower Cost

- 2、统一的平台，支持面向数据、面向事件、面向服务的集成，支持批量、Real Time、同步、异步集成，提供了最大的灵活性和便利性

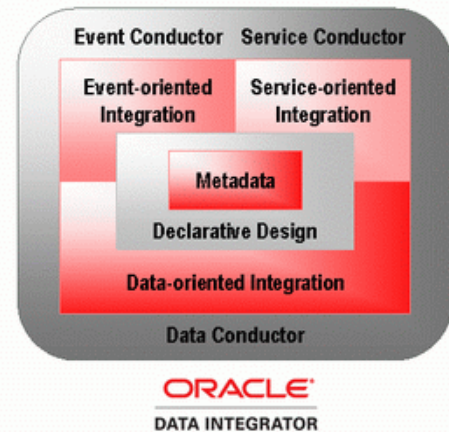
Active Integration Platform Batch, Event-based, and Service-oriented Integration

Active Integration Platform

- Evolve from Batch to Near Real-time Warehousing on Common Platform
- Services Plug into Oracle SOA Suite
- Data Integrity on the Fly

Benefits

- Unify the Silos of Data Integration
- Ensure Overall Data Integrity



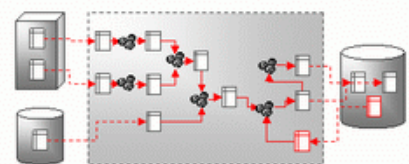
- 3、声明式设计，集成任务无需编程，学习曲线短，工作效率高

Declarative Design Developer Productivity

Conventional Design: Specify Data Flow Graph

- Define every step of complex ETL flow logic
- Require specialized ETL skills and significant development/maintenance efforts

Conventional ETL Design



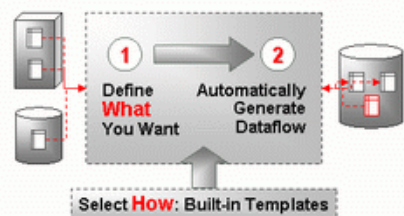
Declarative Set-based Design

- Simplifies the number of steps
- Automatically generates the data flow

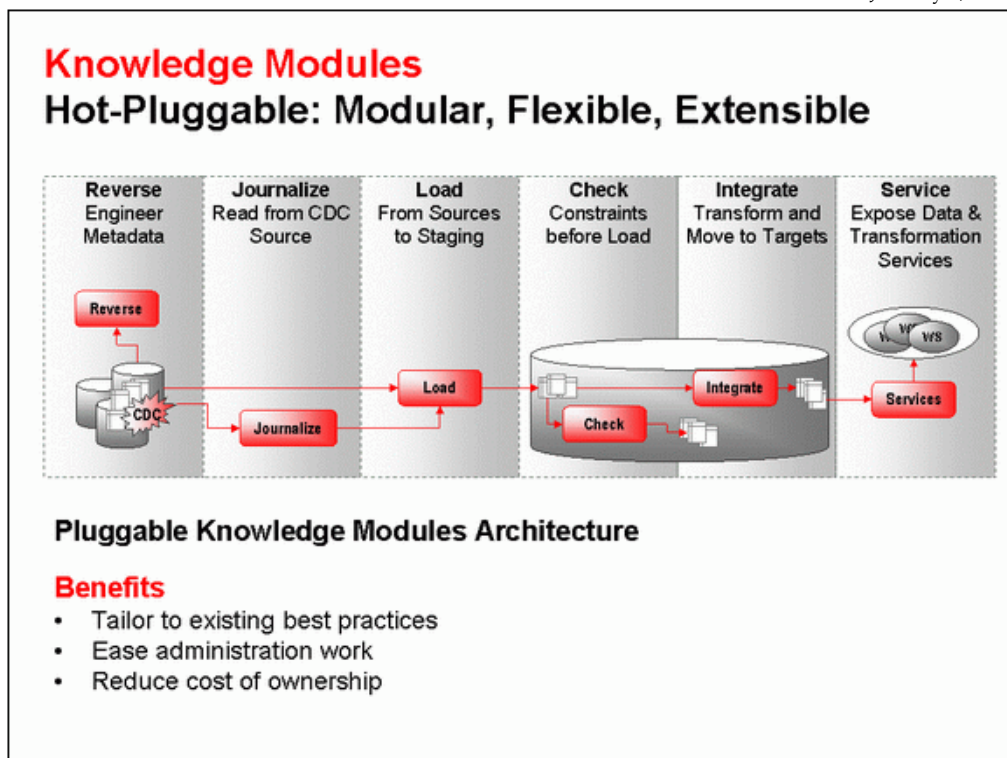
Benefits

- Reduced the learning curve
- Shorter implementation times

ODI Declarative Design

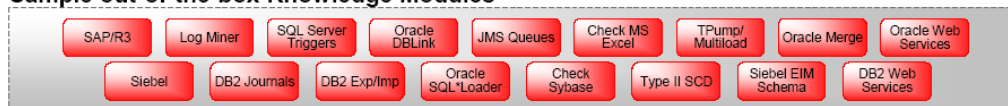


- 4、预置的、可热插入的知识模块，提供了可重用的模块化管理，又确保了灵活性和可扩展性。ODI按阶段将KM分成6类。



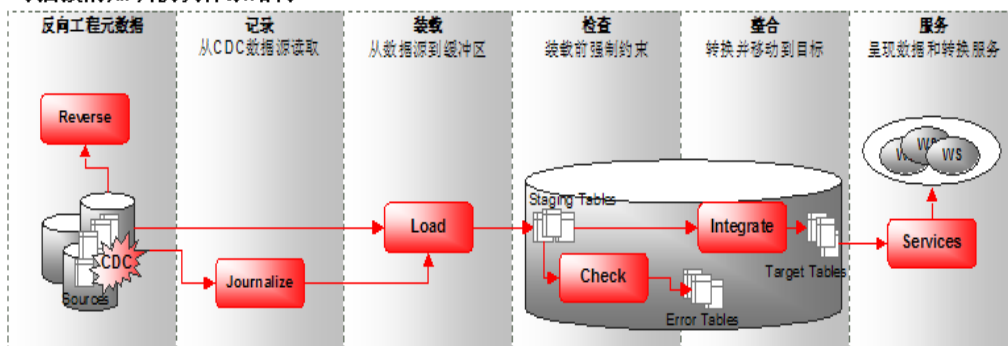
ODI “开箱即用” 的KM有100多个，比如：

Sample out-of-the-box Knowledge Modules

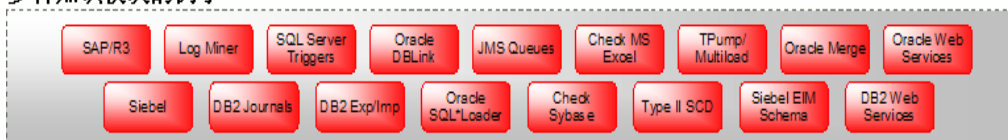


中文版：

可插拔的知识模块体系结构



多种知识模块的例子



1.3. ODI理解之1

1.3.1. 关于ODI与OWB的疑惑

- 1、OWB有DW建模功能，而ODI没有。
- 2、ODI支持更多的数据源，如WS、SOAP、Jython、LDAP Directories。
- 3、ODI支持复杂和实时的数据集成。
- 4、ODI支持将错误数据隔离到专门的Error Table表中而无需编程实现。
- 5、ODI支持CDC、SCD、Bulk Load。

总的来说，在DW领域，ODI是OWB的补充，在纯粹数据集成领域，则应该使用ODI。如果DW项目采用Oracle BIEE，那么最佳的搭档应该就是ODI了。

不过这两个工具今后Oracle有可能合并——ODI 10.1.3.4中已经有了原来属于OWB的Data Profiling and Quality。

1.3.2. 应用场景

任何数据驱动的集成，都可以使用ODI，应用场景包括但不限于：

- 1、数据仓库：比如ETL阶段。ODI+BIEE是非常好的搭配。
- 2、数据迁移：比如将某一源系统的数据迁移到新系统中。
- 3、数据集成：比如两个系统间高效的点到点数据传递。
- 4、数据复制：比如将一个Instance的数据复制另外一个Instance中。
- 5、SOA应用
- 6、MDM应用

1.4. ODI组件

1.4.1. Overview

Oracle Data Integrator由以下基于Java的、可分开部署的组件构成：

- 1、Repository，资料库，分Master Repository和Work Repositories，可安装在RDBMS中。

- 2、Graphical Modules，包括4个设计工具，我们主要用这4个工具工作。

Designer用于定义Data Store、Interface（数据映射）、Package（类似Workflow）。

Operator用于管理和监控数据转换任务的执行情况，也可用于调试。

Topology Manager用于定义物理架构、逻辑架构。

Security Manager用于管理用户权限。

3、Schedule Agent，属于Runtime组件，因为ODI采用E-LT架构，所以Schedule Agent只用来调度执行ELT任务，其数据转换引擎很少用到。

4、Metadata Navigator，基于Servlet和JSP的访问资料库的Web接口。

5、Lightweight Designer，用于通过浏览器查看和编辑Repository。

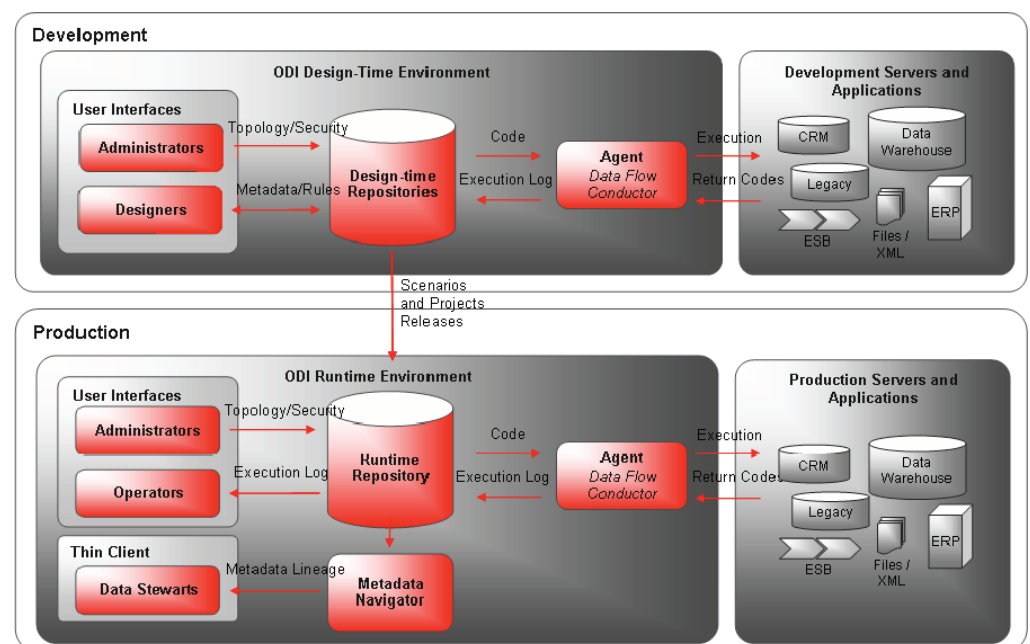
6、Public Web Services，用于SOA环境，可通过WS来访问ODI。

后3个需要单独安装，需要Web服务器如Tomcat或OC4J。

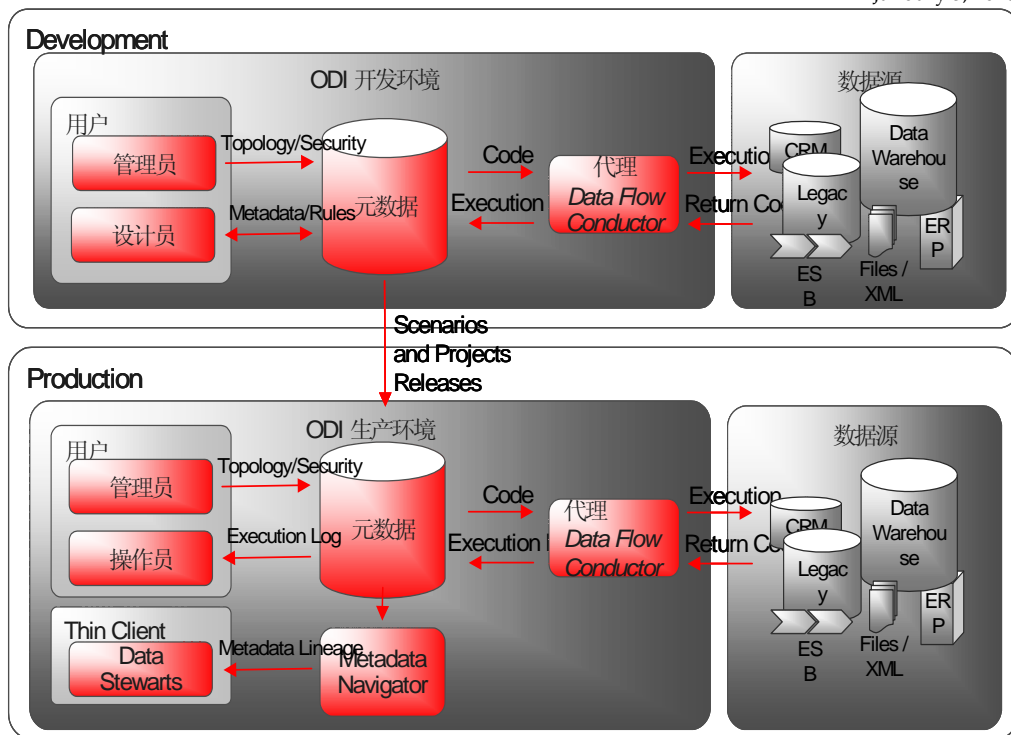
此外，还集成了Oracle Data Profiling、Oracle Data Quality。

1.4.2. 技术架构

开发环境和生产环境：



中文版：



1.5. ODI安装

1.5.1. 软件下载

以10.1.3.4 Windows版本为例，地址：

<http://www.oracle.com/technology/global/cn/software/products/ias/htdocs/101310.html>

1.5.2. 安装选择

- 1、运行安装盘下的setup\Windows\setup.bat。
- 2、产品选择第一个选项安装所有组件——ODI和Data Profiling、Data Quality

选择要安装的产品

☒ Oracle Data Integrator, Oracle Data Profiling, Oracle Data Quality 10.1.3.4.0
Includes Oracle Data Integrator, Oracle Data Profiling and Oracle Data Quality for Oracle Data Integrator.

- 3、类型选择第一个选项同时安装Server和Client

选择安装类型

Oracle Data Integrator, Oracle Data Profiling, Oracle Data Quality 10.1...

您需要何种安装类型？

☒ Complete (1012MB)

Installs client and server components of the Oracle Data Integration Suite.

1.5.3. 安装设置

- 1、Home和路径不要和其它的Oracle产品共用即可

指定主目录详细信息

目标

输入或选择所安装产品的名称，以及安装产品的完整路径。

名称 (N): OUIHome1

路径 (A): C:\OraHome_1

浏览 (B)...

- 2、为Data Profiling和Quality的Server设置端口、管理员及其密码（设为madmin）

Oracle Data Profiling and Quality Server Configuration

The profiling and quality server contains the information for the profiling and quality projects. Please provide configuration information to configure this server.

Repository Port: 7600

Scheduler Port: 7601

Administrator Username: madmin

Administrator Password: *****

注：Windows下netstat -a命令可以查看端口占用情况。

- 3、为Data Profiling和Quality的Client，设置欲连接的主机名和端口

Oracle Data Profiling and Quality Server Connection

The profiling and quality server contains the information for the profiling and quality projects. Please provide the information to connect this server.

Hostname: localhost

Scheduler Port: 7601

Repository Port: 7600

Repository Name: primary

因为Server是本机，所以设置为localhost；端口和上面步骤设置的一样。

- 4、为Data Quality 设置ODBC适配器端口，保持默认

ODBC Adapter Port

Set the port number to use.

ODBC Adapter Port: 7602

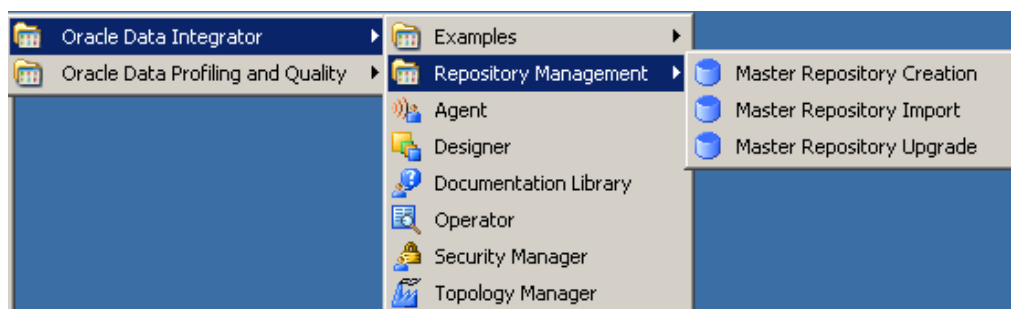
1.6. 服务和菜单

1.6.1. 服务

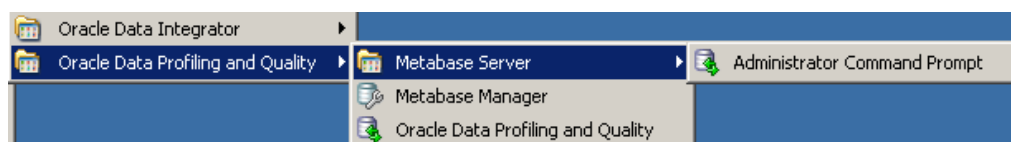
- 1、 Oracle Data Quality Inetd
- 2、 Oracle Data Quality Scheduler

1.6.2. 菜单

- 1、 Oracle Data Integrator



- 2、 Oracle Data Profiling and Quality



1.7. 基本训练

完成官方“Getting Started with an ETL Project”，可比较好的掌握基本概念、功能、集成流程。请勿跳过。

下面的章节将逐一展开，但基本都是记录精要内容，详细地说明请参阅开始菜单中的 Documentation Library。

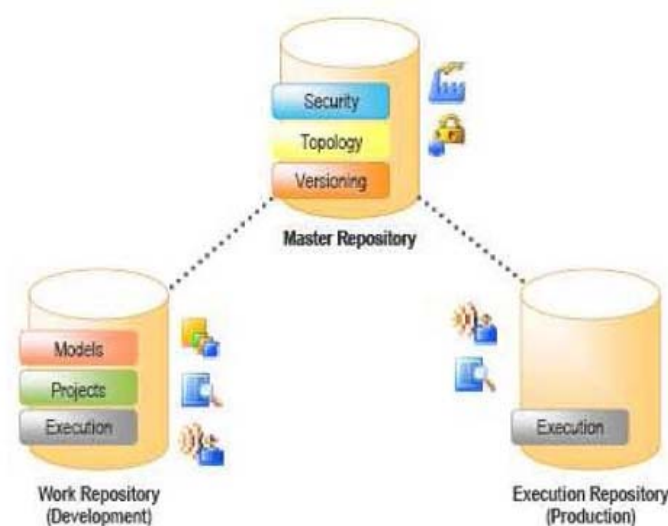
2. 完整的简单例子（资料库、体系结构、项目、模型、接口、包、方案）

2.1. ODI理解之2

2.1.1. 什么是资料库

ODI资料库可安装在任何支持ANSI ISO 89的数据库中。资料库分两种：

- 1、Master Repository，保存企业所有IT资源的Topology，保存项目和数据模型的安全信息、版本信息，供ODI图形模块等使用。通常创建一个即可。Master Repository要尽可能独立存储，单独的Instance，或单独的Schema。
- 2、Work Repository，保存项目和数据模型，供ODI图形模块等使用。可创建多个。一个Work Repository只能连接一个Master Repository。一个Schema只能存储一个Work Repository，不过Master Repository倒可与其安装在同一Schema。



Oracle实验室已验证通过的数据库有：Hypersonic SQL、IBM DB2 UDB、IBM DB2/400、Informix、Microsoft SQL Server、Oracle、Sybase AS Anywhere、Sybase AS Enterprise、Sybase ASIQ等。

2.1.2. 什么是Topology

Topology Manager主要用来管理下面5类任务，并将信息存储在主资料库中，供所有模块共享使用。

- 1、物理体系结构，定义各种技术及其数据服务器、物理架构、物理代理。
- 2、逻辑体系结构，定义各种技术及其关联的逻辑架构、逻辑代理。
- 3、上下文，用来连接物理架构和逻辑架构。
- 4、语言，不同技术所能采用的语言特性。
- 5、资料库，包含主资料库及其附属的工作资料库。

这里重点再看几个概念：

- 1、技术和数据类型：ODI将不同的数据库（Oracle、DB2等）、文件格式（XML File等）、应用系统，用不同的“Technology”来表示。每种技术都有自己支持的“Data Type”。
- 2、数据服务器：在ODI中，每个数据库服务器、JMS消息文件、每组文本文件必须先定义为“Data Server”。
- 3、物理架构：在数据服务器下，每个用户、JMS Topic、文本文件必须定义为一个“Physical Schema”。
- 4、物理代理：在物理体系结构需要定义“Physical Agent”，允许在远程机器上执行ODI任务。
- 5、逻辑架构、逻辑代理是物理架构、物理代理的逻辑组合，方便用户管理和使用。

2.1.3. 什么是知识模块

Oracle Data Integrator之所以能适应不同的、多种多样的数据源，灵活有效的完成数据抽取/转换/载入的过程，均是基于其知识模型体系。Knowledge Modules类似于程序中的插件，Oracle Data Integrator将数据整合的任务抽象出六个组成部分：

- 1、反向工程RKM, Reverse-engineering knowledge modules，用于从数据源读取表及其他对象。
- 2、日记JKN, Journalizing knowledge modules，用于为单一或一组表/视图记录新建的和修改的数据。ODI支持部分数据源的Change Data Capture(CDC)功能，前提为ODI项目中启用该模块。
- 3、加载LKM, Loading knowledge modules，用于从数据源抽取数据。
- 4、检查CKM, Check knowledge modules，用于检测抽取出的源数据的合法性。
- 5、集成IKM, Integration knowledge modules，用于将Staging Area中的数据转换至目标表，基于目标数据库产生对应的转换SQL。
- 6、服务SKM, Service knowledge modules，提供将数据以Web Services的方式展现的功能。

2.2. 环境准备

2.2.1. ODI资料库

资料库采用Oracle Database 10G。请自行安装。假定有如下ORCL DB：

```
ORCL =  
(DESCRIPTION =  
(ADDRESS = (PROTOCOL = TCP)(HOST = HUAJHUA)(PORT = 1522))  
(CONNECT_DATA =  
(SERVER = DEDICATED)  
(SERVICE_NAME = orcl)
```

```
)  
)
```

如果是9i之前，因为不支持char类型，需要将\lib\scripts\xml\TECH_Oracle.xml中的VARCHAR2(%L CHAR)替换为VARCHAR2(%L)。

2.2.2. 源系统

源系统采用Oracle Database 10G自带的scott.emp、scott.dept，简单起见也用ORCL。不过scott用户需要解锁，并且需要授权：

```
alter user SCOTT account unlock;  
alter user scott identified by tiger;  
grant connect,resource to scott;  
grant create view to scott;
```

2.2.3. 目标系统

目标系统也是Oracle Database 10G，简单起见也用ORCL，用户名ODI_TRG。

```
create user ODITRG identified by ODITRG default tablespace  
users;  
grant connect,resource to ODITRG;  
grant create database link to ODITRG;  
grant create synonym to ODITRG;
```

```
create table ODITRG.EMP  
(  
    EMPNO      NUMBER(4) not null,  
    ENAME      VARCHAR2(10),  
    JOB        VARCHAR2(9),  
    MGR        NUMBER(4),  
    HIREDATE   DATE,  
    SAL        NUMBER(7,2),  
    COMM       NUMBER(7,2),  
    DEPTNO     NUMBER(2)  
);
```

```
create table ODITRG.DEPT  
(  
    DEPTNO     NUMBER(2) not null,  
    DNAME      VARCHAR2(14),  
    LOC        VARCHAR2(13)  
);  
alter table ODITRG.DEPT  
    add constraint PK_DEPT primary key (DEPTNO);
```

2.3. 创建资料库

2.3.1. 创建用户

创建Master Repository用户：

```
create user snpm identified by snpm default tablespace users;  
grant connect, resource to snpm;
```

创建Work Repository用户:

```
create user snpw identified by snpw default tablespace users;  
grant connect, resource to snpw;
```

注：也允许这两个资料库用户在不同的服务器上。


2.3.2. 创建Master Repository

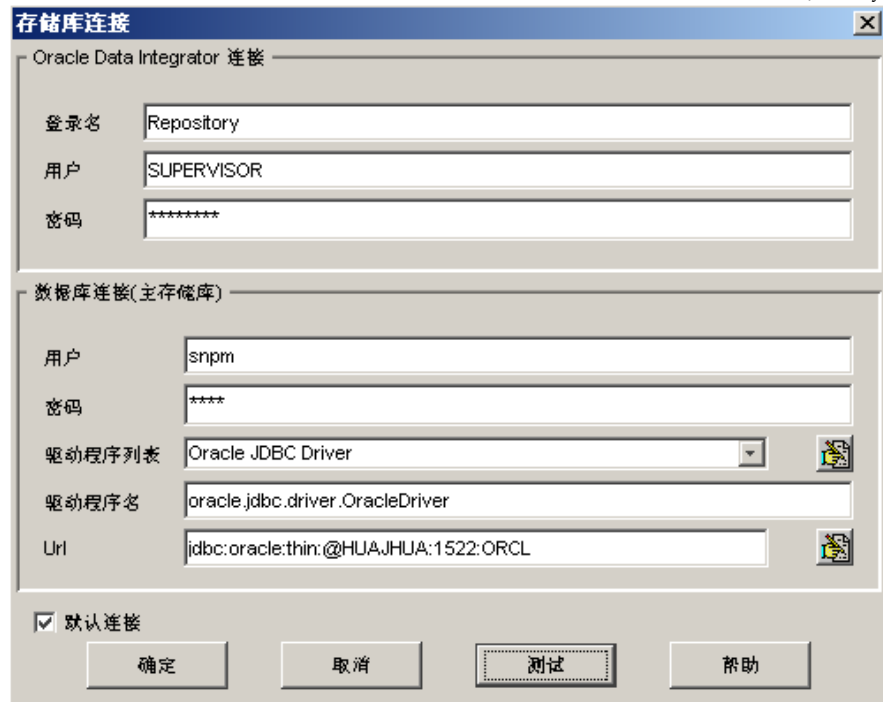
通过开始菜单Master Repository Creation启动，并按下图输入资料库信息（密码为snpm）：



通过“测试连接”可测试配置是否正确。“确定”后开始创建数据库对象。

2.3.3. 连接Master Repository（创建存储连接“Repository”）

通过开始菜单Topology Manager启动，点击新建按钮“”，并按下图输入资料库信息（SUPERVISOR密码为SUNOPSIS）：



JDBC驱动程序: oracle.jdbc.driver.OracleDriver

JDBC URL: jdbc:oracle:thin:@HUAJHUA:1522:ORCL

通过“测试”可测试配置是否正确，如果密码过于简单或与用户名相同，会报“密码无效”错误。“确定”后会到登录界面：

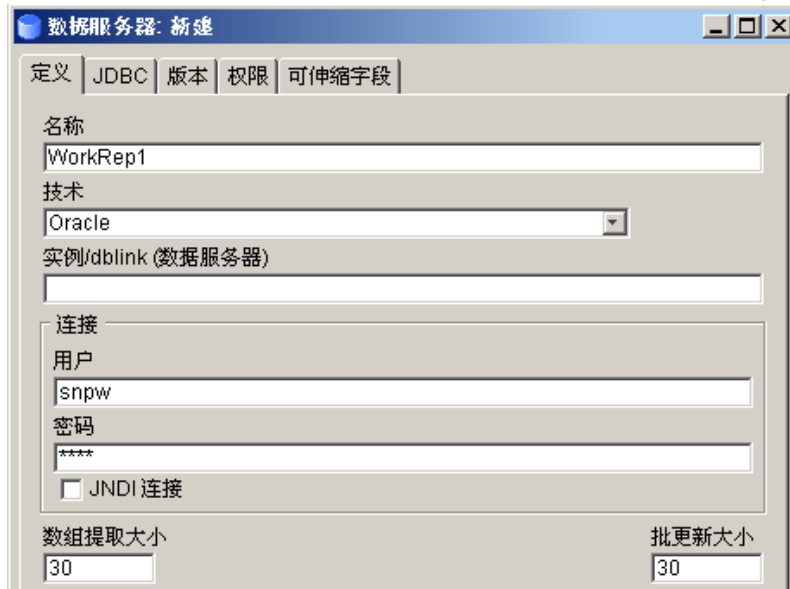


2.3.4. 创建Work Repository

N:Topology Manager\资料库\工作资料库\右键\插入工作资料库



在定义Tab页，按照下图输入连接名称、技术、用户和密码：



在JDBC Tab页，输入如下信息：

JDBC驱动程序: `oracle.jdbc.driver.OracleDriver`

JDBC URL: `jdbc:oracle:thin:@HUAJHUA:1522:ORCL`


通过“测试”测试配置是否正确。

“确定”后再随后的界面，输入如下信息：



“确定”后完成Work Repository创建。

2.3.5. 连接Work Repository（创建工作存储连接“Repository”）

通过开始菜单Designer启动，点击新建按钮“”，并按下图输入资料库信息（SUPERVISOR密码为SUNOPSIS）：

工作存储库连接

Oracle Data Integrator 连接

登录名: Repository

用户: SUPERVISOR

密码: *****

数据库连接(主存储库)

用户: snpm

密码: ****

驱动程序列表: Oracle JDBC Driver

驱动程序名: oracle.jdbc.driver.OracleDriver

Url: jdbc:oracle:thin:@HUAJHUA:1522:ORCL

工作存储库

存储库名称: WORKREP1

☒ 默认连接

确定 取消 测试 帮助

JDBC驱动程序: `oracle.jdbc.driver.OracleDriver`

JDBC URL: `jdbc:oracle:thin:@HUAJHUA:1522:ORCL`

通过“测试”可测试配置是否正确，如果密码过于简单或与用户名相同，会报“密码无效”错误。“确定”后会到登录界面：

Oracle Data Integrator 登录

登录名: Repository

用户: SUPERVISOR

密码: *****

确定 取消 帮助

“确定”后就进入Designer模块了。

2.4. 创建物理体系结构

2.4.1. 创建数据服务器ORCL_SCOTT

N:Topology Manager\物理体系结构\技术\Oracle\右键\插入数据服务器

1、定义数据服务器名称、Dblink、连接用户名和密码：



注1：在“Oracle to Oracle”模式时，“实例/DBLINK”将用于ODI自动创建两个系统间的DBLINK，为了使DBLINK创建成功，需要在目标数据库端配置TNSNAME（=这里的“实例/DBLINK”）；另外如果来源和目标是同一个数据库，那么DBLINK就是LOOPBACK的，这里的“实例/DBLINK”名字就不能等于SID。故这里故意加了“.LOOPBACK”。

注2：这个定义界面的用户名和密码，用于统一连接这个数据服务器，通常它的权限比较大，能够访问很多其他用户的数据。而下面的“物理架构”里面选择的用户名，不是用来连接的，只是一个具体的用户。ODI要求每个用户单独设置一个“物理架构”。

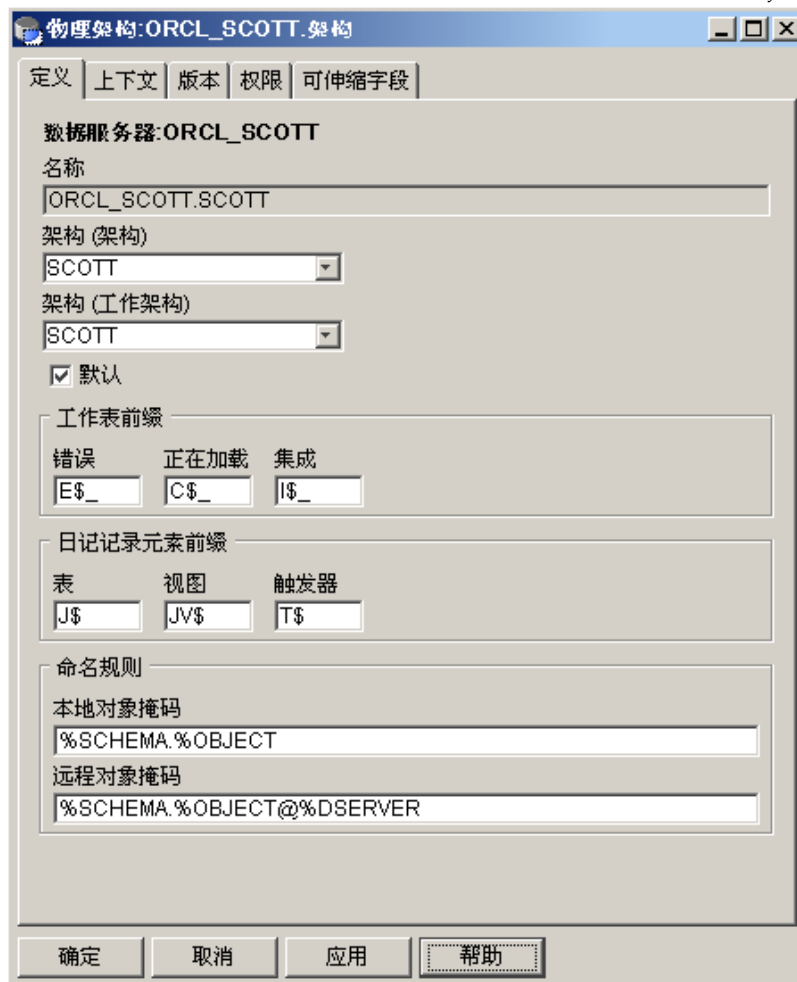
2、定义JDBC

驱动程序 `oracle.jdbc.driver.OracleDriver`

URL `jdbc:oracle:thin:@HUAJHUA:1522:ORCL`

2.4.2. 创建物理架构ORCL_SCOTT.SCOTT

N:上面步骤确定后，会自动弹出创建物理架构的界面：



这里需要选择两个架构（Schema），第一个是Data Schema。第二个是Work Schema，什么意思呢？ELT中的T，需要创建一些工作表等临时对象如错误数据表、视图等，这些对象需要存储在工作Schema下。简单起见这里都选择SCOTT。

其他参数默认即可。确定后报没有选择上下文的警告，先忽略之。

2.4.3. 创建数据服务器ORCL_ODITRG

以同样步骤，创建数据服务器ORCL_ODITRG、物理架构ORCL_ODITRG. ODI TRG

结果如下：



驱动程序 oracle.jdbc.driver.OracleDriver

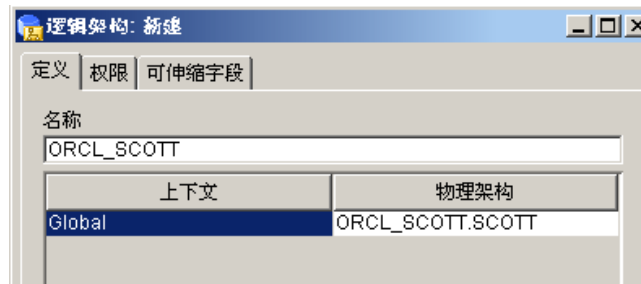
URL jdbc:oracle:thin:@HUAJHUA:1522:ORCL

2.5. 创建逻辑体系结构

2.5.1. 创建逻辑架构ORCL_SCOTT

N:Topology Manager\逻辑体系结构\技术\Oracle\右键\插入逻辑架构

按下图录入命名和上下文，确定后在物理架构那里也会自动添加上下文。



2.5.2. 创建逻辑架构ORCL_ODITRG

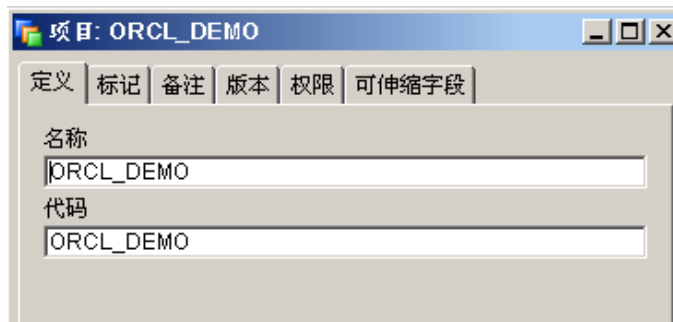
步骤同上。结果如下：



2.6. 创建项目

2.6.1. 创建项目ORCL_DEMO

N:Designer\<项目名>\右键\插入项目，输入名称ORCL_DEMO



2.6.2. 导入知识模块

N:Designer\<项目名>\ORCL_DEMO\知识模块\RKM\右键\导入知识模块



文件导入目录选择 “C:\OraHome_1\oracledi\impexp”



可以逐个选择需要的知识模块，为简单起见，全选后确定。不过全部导入比较耗时和占空间，看下面的滚动条和右下脚的空间占用就知道了。

2.7. 创建模型

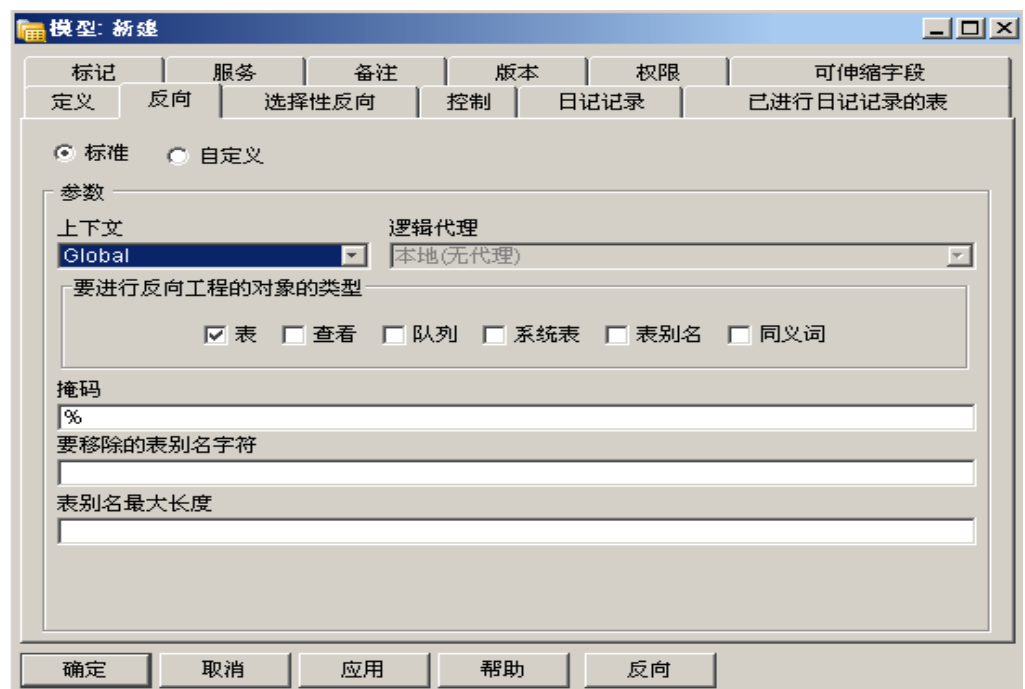
2.7.1. 创建模型ORCL_SCOTT

N:Designer\模型\右键\插入模型

1、按下图内容输入名称、技术、逻辑架构



2、在反向标签页，上下文选择“Global”，然后点击“反向”按钮



这样，就把Scott下的表，全部反向工程到我们的模型中——BONUS、DEPT、EMP、SALGRADE:



2.7.2. 创建模型ORCL_ODITRG

以同样的逻辑创建ORCL_ODITRG。不过要添加个约束，以便后面的接口能够使用默认的CKM。

N:\Designer\模型\ORCL_ODITRG\EMP\约束\右键\插入键

名称: PK_EMPNO_1

键或索引类型: 主键

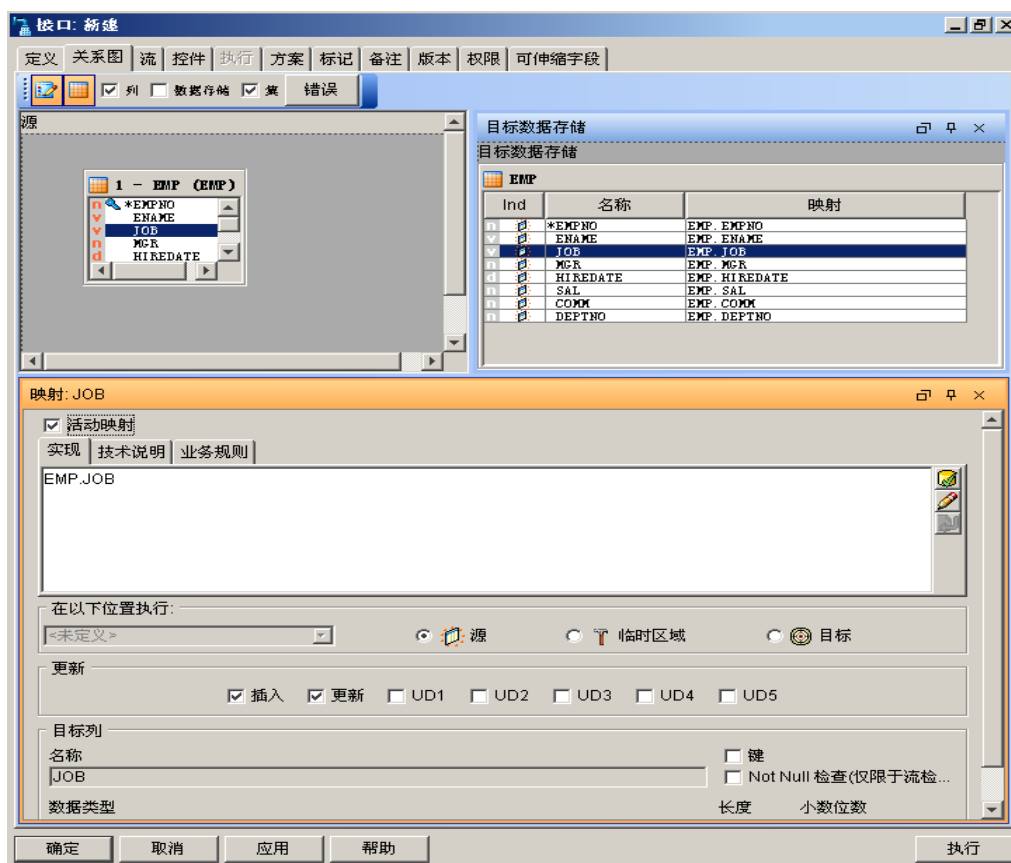
列: EMPNO

2.8. 创建接口

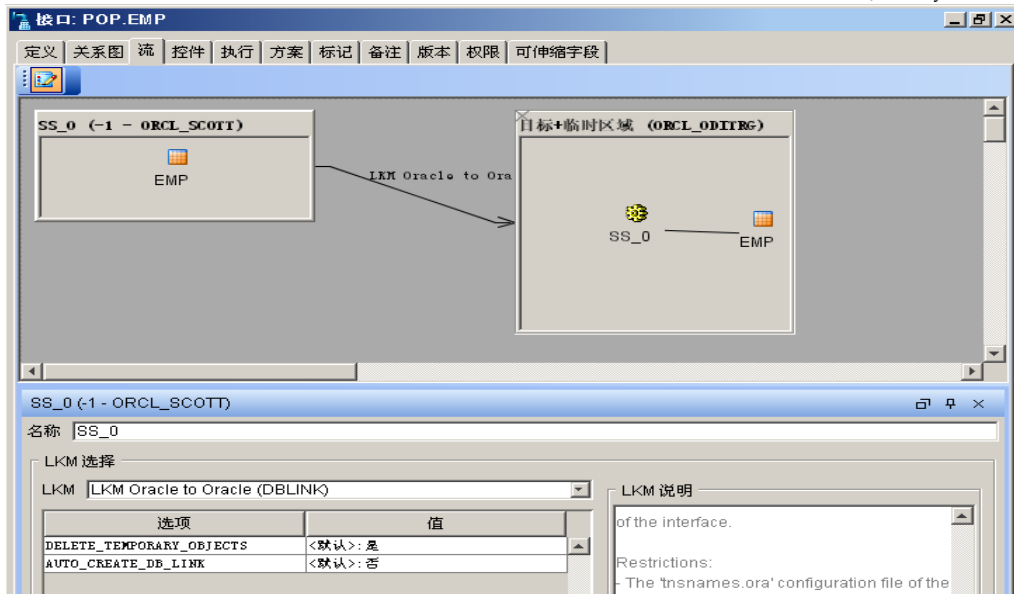
2.8.1. 创建接口POP.EMP

N:\Designer\<项目名>\ORCL_DEMO\<文件夹名>\接口\右键\插入接口

- 1、在“定义”标签页，输入名称: POP.EMP
- 2、在“关系图”标签页，将ORCL_TRG模型下的EMP拖到“目标数据存储”这个区域；将ORCL_SCOTT模型下的EMP拖到“源”这个区域，并让Designer自动映射，结果如下：

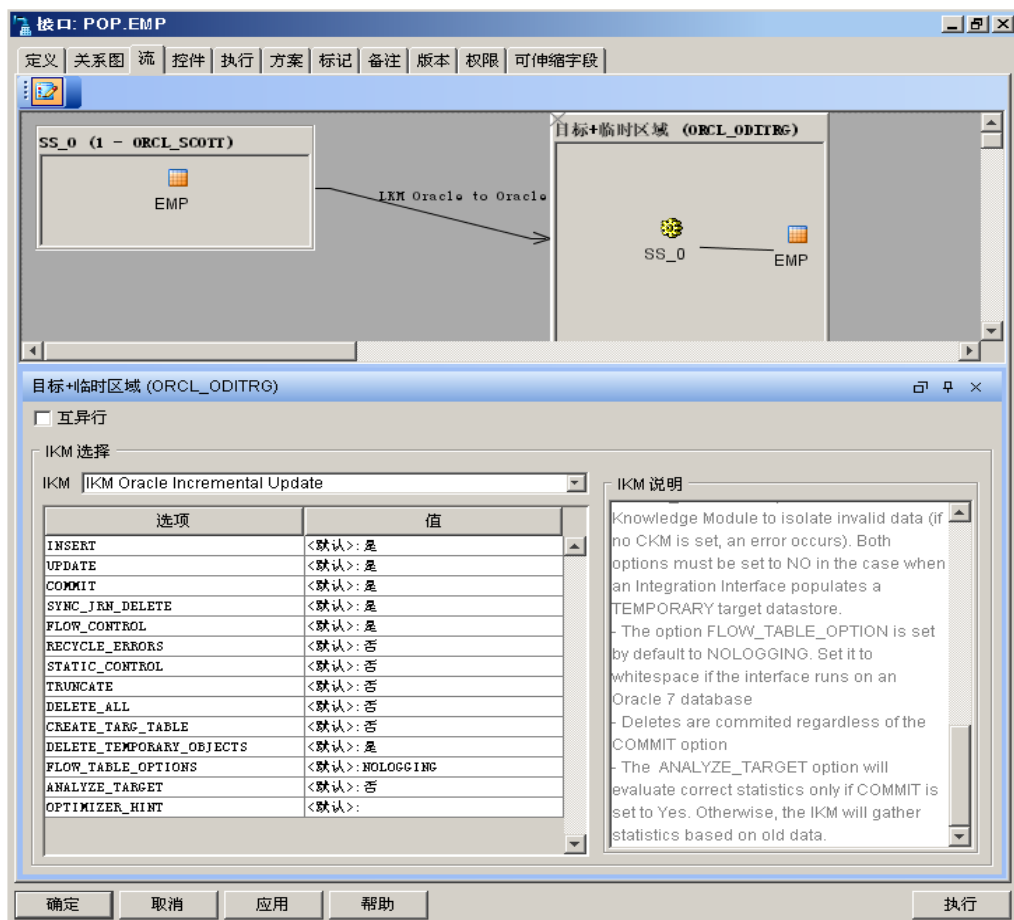


- 3、切换到“流”标签页

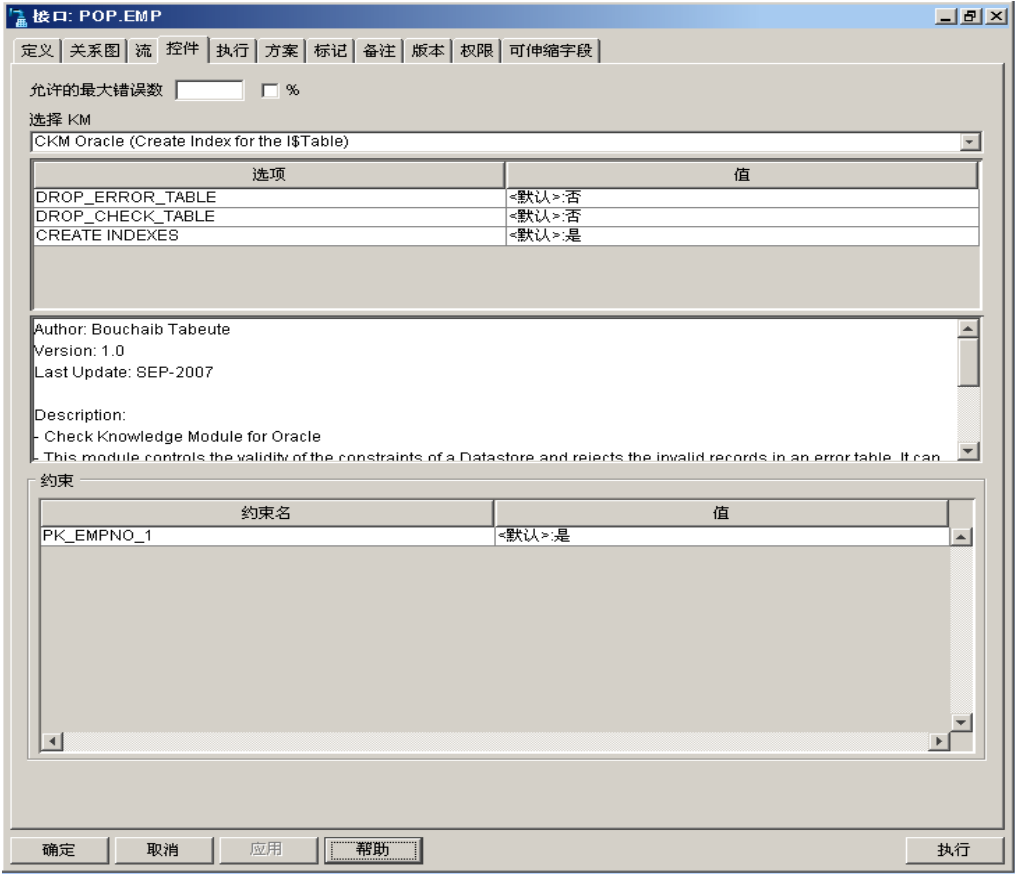


选中SS_0, LKM已默认采用LKM Oracle to Oracle (DBLINK), 我们需要把下面的AUTO_CREATE_DB_LINK改为是。注, 如果我们已经在数据库中手工创建了DBLINK, 并且名字和定义“数据服务器”时定义的一样, 这里就保持默认值否。

选中“目标+临时区域”, 可以看到下面已经默认使用了IKM Oracle Incremental Update



4、切换到“控制”标签页



把默认的CKM改为CKM Oracle。

确定后退出“接口”定义界面。

2.9. 运行接口

2.9.1. 运行接口

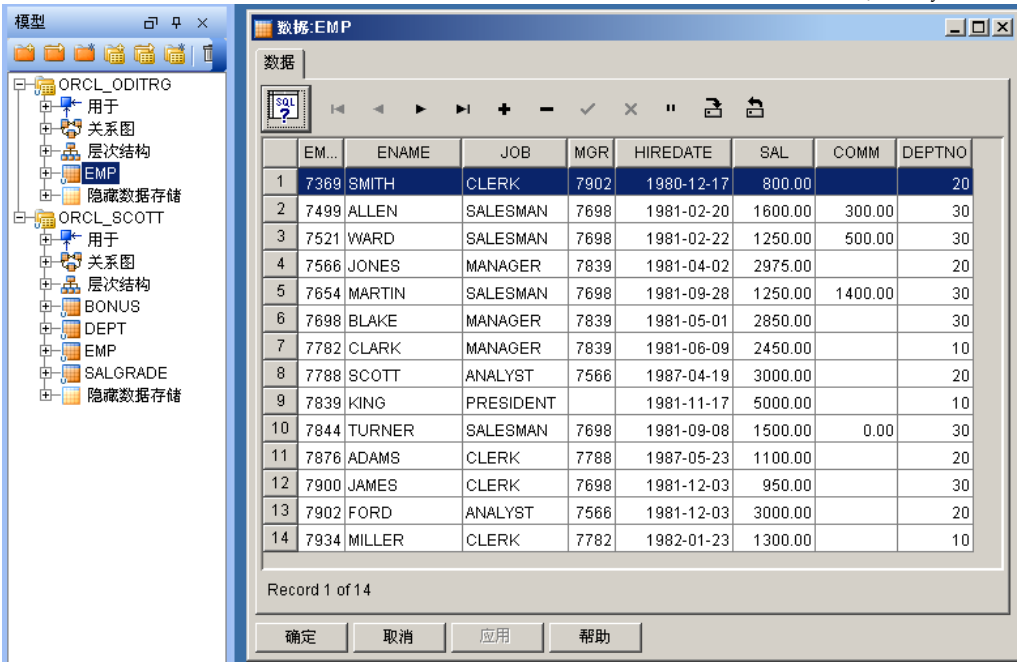
N:Designer\<项目名>\ORCL_DEMO\<文件夹名>\接口\POP.EMP\右键\执行



我们不使用代理，保持上面的选项点击“确定”，将会启动ELT会话。

2.9.2. 查看数据

N:Designer\模型\ORCL_ODITRG\EMP\右键\数据

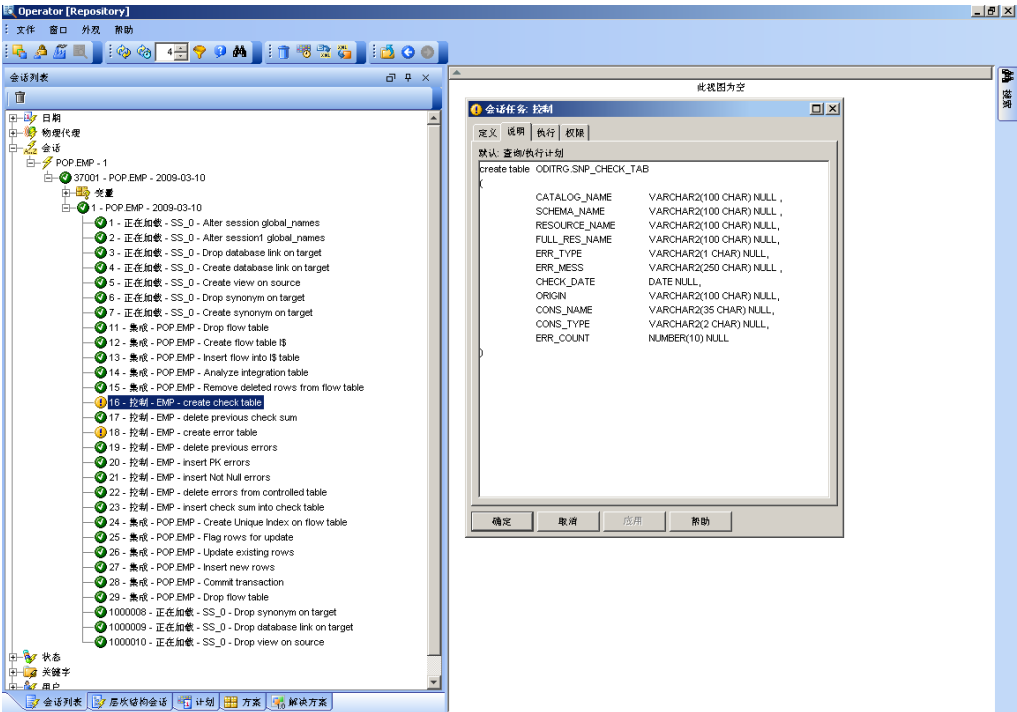


如果一切顺利，就可以看到上面的结果，数据已经从ORCL_SCOTT传到ORCL_TRG了。

2.10. 监控和查看会话状态

2.10.1. 启动Operator

N:Operator，选择Designer一样的登录连接“Respository”进行登录



上面是刚才正常结束的接口POP.EMP执行会话。

2.10.2. 问题诊断

如果出现问题，Operator也提供了比较好的调式方法：

- 1、 点击出错的结点，在“说明”标签页，可以看到要执行的代码
- 2、 在“执行”标签页，可以看到执行的结果和具体的错误信息
- 3、 对于出错的会话，可以重新执行，并且，可以先修改“说明”标签页的内容，加入调式信息

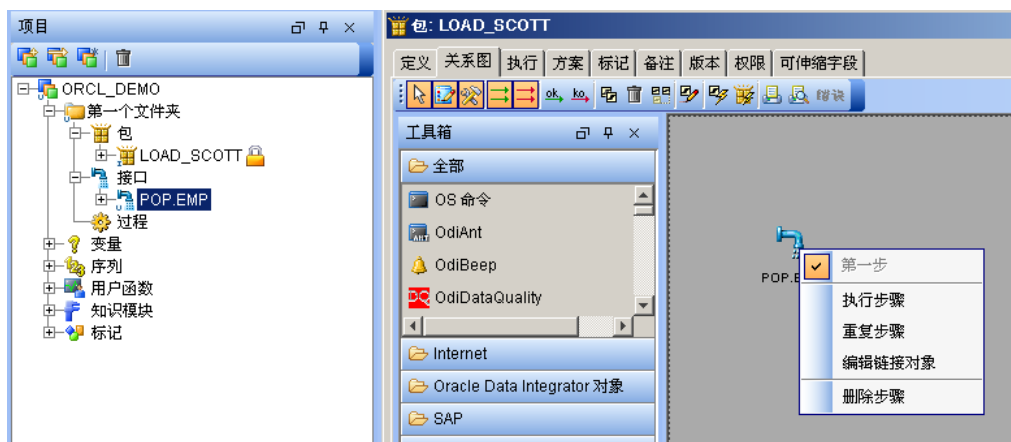
2.11. 创建并运行包

2.11.1. 创建包LOAD_SCOTT

N:\Designer\<项目名>\ORCL_DEMO\<文件夹名>\包\右键\插入包

名称：LOAD_SCOTT

切换到“关系图”标签页，把“接口”下的“POP.EMP”拖进来：



结果如上图，当右键POP.EMP，可以看到，“第一步”以自动选中了。

2.11.2. 运行包LOAD_SCOTT

N:\Designer\<项目名>\ORCL_DEMO\<文件夹名>\包\LOAD_SCOTT\右键\运行

切换到Operator，可以看到会话正常完成：



2.12. 创建并运行方案

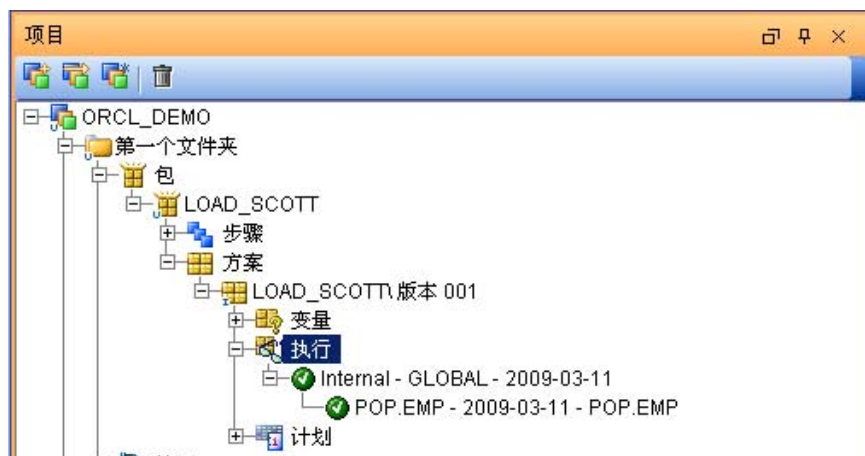
2.12.1. 生成方案LOAD_SCOTT 001

N:\Designer\<项目名>\ORCL_DEMO\<文件夹名>\包\LOAD_SCOTT\右键\生成方案



2.12.2. 运行方案LOAD_SCOTT 001

N:\Designer\<项目名>\ORCL_DEMO\<文件夹名>\包\LOAD_SCOTT\方案\LOAD_SCOTT\版本001\右键\执行



在方案这里，执行完可以直接查看运行状况，如上图，不需要切换到Operator

2.13. ODI理解之3

基于以上例子，做个小结。

2.13.1. 设计过程

- 1、[Master Repository Creation]创建主资料库，一般1个即可
- 2、[Topology Manager]创建工作资料库，可以创建多个，也可以仅创建1个
- 3、[Topology Manager]创建数据服务器，每个Instance或者应用创建一个，底下按照用户创建不同的物理架构

- 4、[Topology Manager]创建逻辑架构，通常与物理架构一一对应，也可以一个逻辑架构对应多个物理架构
- 5、[Designer]创建项目，导入知识模块，或自己开发知识模块
- 6、[Designer]创建模型，每个模型对应到逻辑架构
- 7、[Designer]创建接口，修改必要的知识模块和选项
- 8、[Designer]运行接口
- 9、[Operator]查看和监控运行结果，包括出错脚本和信息
- 10、进一步可创建包、方案，并运行

2.13.2. 几个关系

1、接口、包、方案

简单的话，如我们上面的例子，接口、包、方案的内容和运行结果一样。实际上，接口是ELT基本单元，包类似工作流，用于将接口串联起来，方案是对象的发布版本，可理解为预编译的。

2、6个KM

RKM用于在模型中进行反向工程，把数据库的表定义、约束等导入到ODI中。

SKM用于将接口等发布为Web Service。

运行期ODI按照交替执行其他KM，JKM、LKM、IKM、CKM。

2.13.3. 方案

方案是对象的发布版本，可理解为预编译的。

Variable变量、Procedure过程、Interface接口、Package包都可以发布为方案，方案。方案可以在Designer中运行，也可以通过Operator、操作系统命令、Web Service、HTTP URL执行。

此外，方案可以在Operator中导入、导出，可用于发布到正式环境。

3. 最常用特性和功能实例一（CDC、Agent、Schedule）

上面的例子虽然极其简单，但演示了整个过程，至少可以初学者迈出第一步；接下来我们看看一些其他重要的功能，至于基本开发过程，就不再详细说明了。

3.1. ODI理解之4

3.1.1. 丰富的数据约束

与数据库中的约束一样，是为了保证数据的一致性。实际上这些约束可以直接在RDBMS中创建。而ODI中的约束则是保存在资料库中的，对书DB无影响。

ODI支持如下约束类型：关键字（主键、普通索引、唯一性索引）、引用、约束条件。并且，也是创建在“表”上的。

约束的控制有两种：对已有的数据进行“Static Control”静态检查，对传递的数据进行“Flow Control”动态检查。

在源系统符合约束条件的数据，未必符合目标系统的约束——目标系统可能要求更高，而源系统缺少约束或者没必要约束。

那么对于违反约束的“错误”数据怎么办，ODI怎么处理呢？ODI将其隔离在Error表中。每个目标表都有一个错误表（表名为E\$_<目标表>），可通过“Designer/模型/<目标表>/控制/错误”来查看和编辑错误数据。这就是ODI的“Firewall”——垃圾数据不进入目标系统。

3.1.2. 灵活的执行环境

ODI强大的功能，部分来自于其ELT中T的执行环境比较灵活，可以是：

源系统Source 、目标系统Target 、临时区域Staging Area 。

在哪里执行效率高就在哪，充分发挥各个系统的性能优势。

如果某个环境不支持所要求的“T”，Interface的Diagram定义界面的Error按钮就会亮起来，点击可查看具体错误。

3.2. ODI理解之5（CDC）

3.2.1. CDC概述

已更改数据捕获的英文全称是Changed Data Capture(CDC)。

CDC用于记录源数据的变动情况到日记表，这样在传递数据时，不用考虑未变动的数据，大大提高ELT的效率。类似Oracle的快速刷新型MV。

ODI CDC能够把源Insert、Update、Delete同步到目标。

CDC分两种类型：

- 1、简单（Simple）：通常用于单个Datastore（通常就是代表一个表）。
- 2、一致性集（Consistent Set）：当一组Datastore采用CDC方式时，为保证数据的前后顺序和完整性制约关系，我们需要告诉ODI按照一定的顺序来抓取，ODI是通过设置Consistency Window来实现的。这样一组Datastore就叫做Consistent Set。此外，ODI建议，如果Subscriber很多，用Consistent Set有助于提高性能。

注：对于一个Datastore来说，Simple和Consistent Set不能同时存在。

3.2.2. CDC工作原理

CDC的工作原理：通过在源表自动创建触发器（T\$开头）或者通过源数据库的LOG挖掘，得到净DML变更数据的主键，放到ODI创建的J\$日记表中，并通过JV\$日记视图提供完整的变更数据，供ELT直接使用。ODI把这个叫做“Journalizing Models”。

Journalizing Models由以下几个部分组成：

- 1、Journals日记：用来记录源数据DML变更，其实就是一些J\$开头的表
- 2、Capture捕获：通过在表上自动创建的触发器或Log挖掘相关的Package，获得变更数据，放入Journals中
- 3、Subscribers订阅者：CDC采取Publish/Subscribe模式，Capture到的Journal相当于Publish，需要有Subscriber来订阅和消费。对于ODI来说，如果没有订阅者，那么就不执行Capture；如果所有的Subscriber都执行了消费，Journal的数据就不在保留。
- 4、Journalizing Views日记视图：ODI提供了一组jv\$视图，用于IKM执行时获取变化的数据，当然了我们也可通过SQL直接察看变化的数据。

3.2.3. Journalizing Infrastructure

CDC将根据物理Schema的定义，分别在Data Schema和Work Schema创建如下对象：

- 1、T\$触发器，创建在每个Data Schema的表下
- 2、SNP_CDC_表，这个是CDC的通用表，创建在标志为Default的的物理Schema的Work Schema下，所以同一Data Server，只需创建一次，因为只有1个Default的的物理Schema。
- 3、J\$表、JV\$视图，创建在每个Work Schema下。

3.2.4. CDC小钟图标

- 1、黄色：Journalizing已启用，但Infrastructure还没创建好，通常是刚配完，未启动
- 2、绿色：Journalizing正常，一切就绪。
- 3、灰色：Journalizing已失效，通常是Datastore被移出CDC，但Journalizing没停

3.3. O2O CDC (Simple)

以Dept、EMP为例。

3.3.1. 完成普通设置步骤

参照前面Oracle to Oracle，完成模型创建，比如DEPT、EMP。

3.3.2. 模型日记记录设置

N:Designer\模型\<源模型名，如ORCL_SCOTT>\双击\日记记录

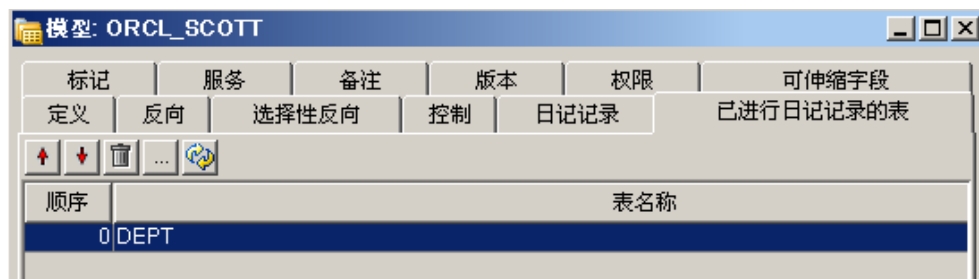



ODI已经默认选择模式为“简单”、KM为“JKM Oracle Simple.ORCL_DEMO”。

3.3.3. 添加到CDC

N:Designer\模型\ORCL_SCOTT\<Datastore如DEPT>\右键\已更改数据捕获\添加到CDC

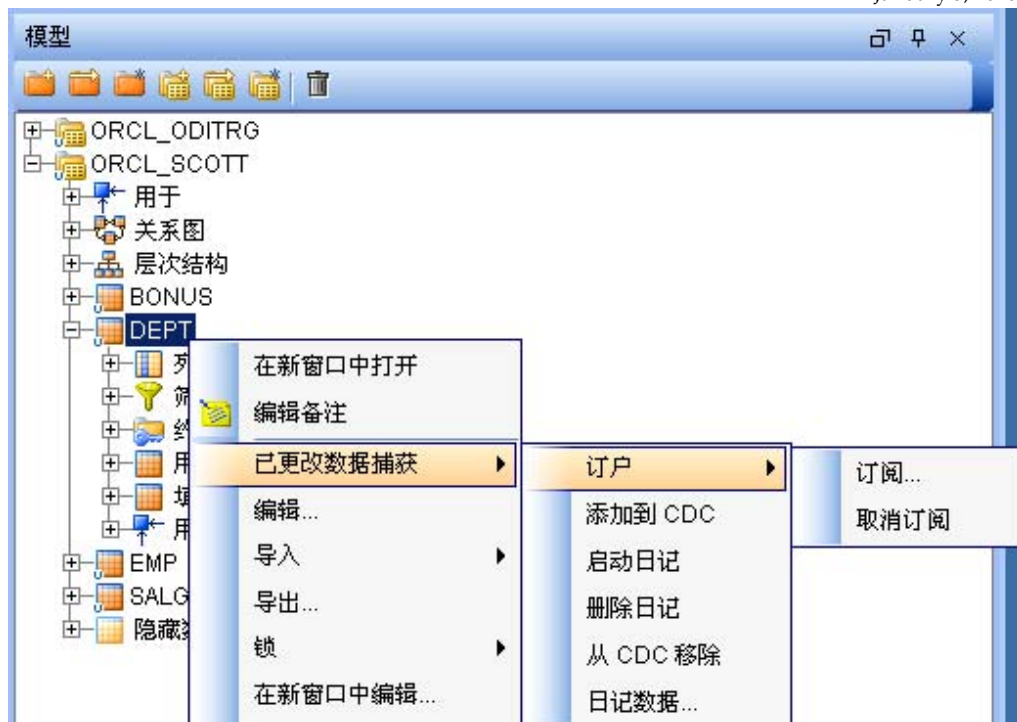
添加到CDC后可以看到有个变化：模型ORCL_SCOTT的属性“已进行日记记录的表”



另外，Datastore的图标上，左上角多了个黄色的小钟： DEPT。

3.3.4. 增加订阅

N:Designer\模型\ORCL_SCOTT\DEPT\右键\已更改数据捕获\订户\订阅



增加一个名字叫“DEPT”的订阅者。名字可任意，后面设置接口的Filter时要用到。

数据存储DEPT的属性“日记记录”的变化：



3.3.5. 启动日记

N:Designer\模型\ORCL_SCOTT\DEPT\右键\已更改数据捕获\启动日记

注：增加订阅、增加订阅这两个步骤，也可以在Package中设置，这样方便移植。操作很简单，把Datastore或整个Model拖到Package的Diagram中，Type选择Journalizing Datastore或Model。

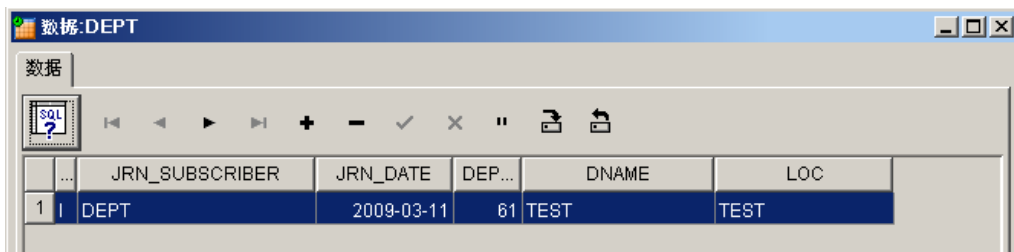
注：If a datastore with journals running is removed from the CDC in simple mode, the journals should be stopped for this individual datastore.

3.3.6. 测试

1、在PL/SQL Developer中执行下列语句往源系统插入数据

```
INSERT INTO scott.dept  
SELECT MAX(deptno) + 1, 'TEST', 'TEST' FROM scott.dept;
```

- 2、N:<源Data Store如ORCL_SCOTT下的DEPT>\右键\已更改数据捕获\日记数据，可以看到，日记中有一条新增记录：



	JRN_SUBSCRIBER	JRN_DATE	DEP...	DNAME	LOC
1	DEPT	2009-03-11	61	TEST	TEST

3.3.7. 创建普通接口

注：加了CDC功能的Data Store对普通的接口没有任何影响，接口跑完，该过去的的数据都会过去，已更改数据捕获\日记数据中的数据依然存在。

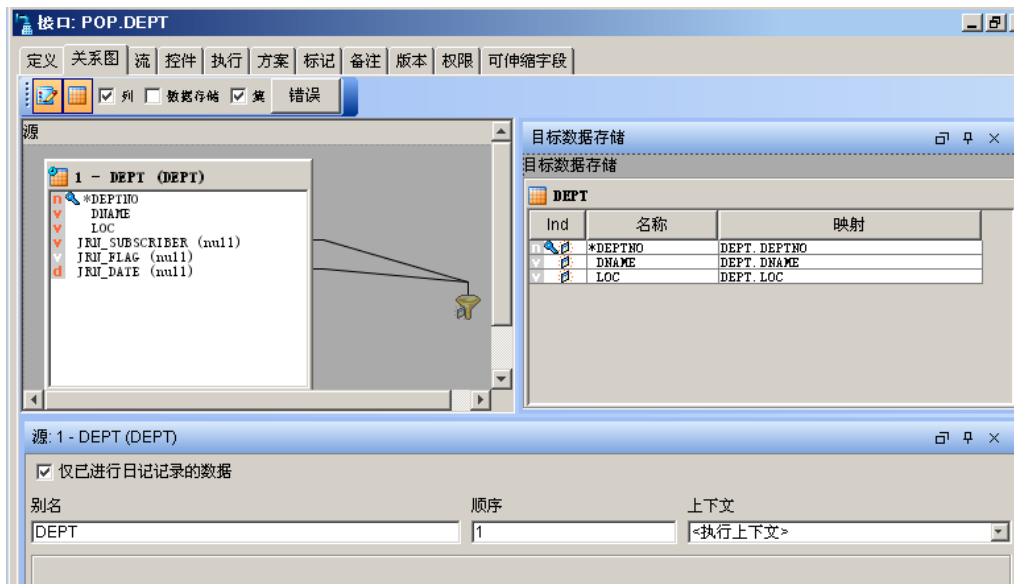
参照前面Oracle to Oracle，接口名为POP.DEPT_O2O_SIMPLE，完成接口创建、同步数据。


- 1、N:<目标Data Store如ORCL_ODITRG下的DEPT>\右键\数据，可以看到所有数据
- 2、N:<源Data Store如ORCL_SCOTT下的DEPT>\右键\已更改数据捕获\日记数据，记录还在！

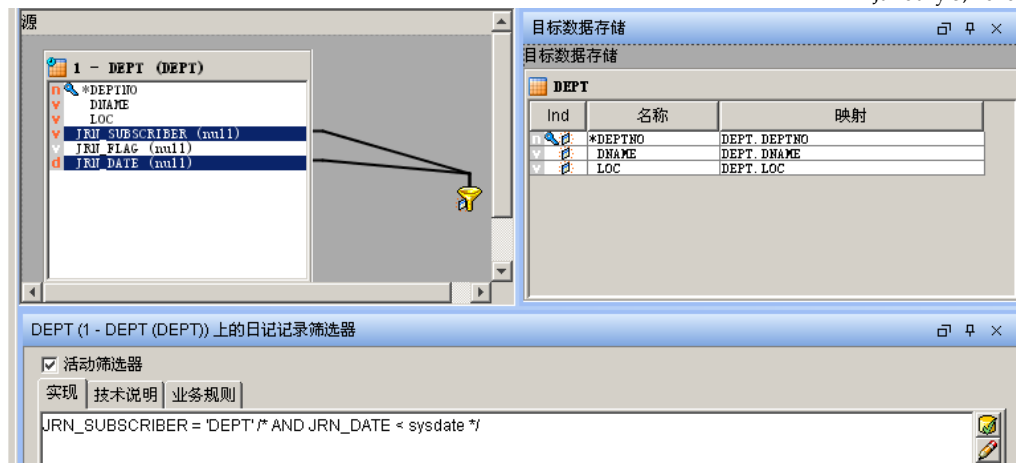
3.3.8. 接口启用CDC

N:Designer\<项目名>\<文件夹名>\接口\POP.DEPT_O2O_SIMPLE\双击\关系图

选中源“1 - DEPT (DEPT)”，勾选下面的“仅已进行日记记录的数据”：



选中漏斗，修改下面的筛选条件为JRN_SUBSCRIBER='DEPT'：



3.3.9. 运行接口

N:\Designer\<项目名>\<文件夹名>\接口\POP.DEPT_O2O_SIMPLE\右键\运行

3.3.10. 数据验证

1、在PL/SQL Developer中，再执行下列语句往源系统插入数据

```
INSERT INTO scott.dept
SELECT MAX(deptno) + 1, 'TEST', 'TEST' FROM scott.dept;
```

2、N:<源Data Store如ORCL_SCOTT下的DEPT>\右键\已更改数据捕获\日记数据，可以看到，日记中有一条新增记录。

3、执行启用Simple CDC的接口，如POP.DEPT_O2O_SIMPLE

4、N:<目标Data Store如ORCL_ODITRG下的DEPT>\右键\数据，可以看到新增数据

5、N:<源Data Store如ORCL_SCOTT下的DEPT>\右键\已更改数据捕获\日记数据，记录不见了

问题：如果有两个Interface，筛选条件为JRN_SUBSCRIBER='DEPT'，那么第一个Interface跑完，第二个岂不是没数据可抓了？

结论：如果有多个Interface要用到同一个CDC，那么就要定义不同的订阅。

3.3.11. 配置EMP参

照上面步骤，配置EMP的SIMPLE CDC，接口名为POP.EMP_O2O_SIMPLE。

造数据可用如下脚本：

```
INSERT INTO scott.emp
SELECT MAX(empno) + 1, MAX(ename), MAX(job), MAX(mgr), MAX(hiredate), MAX(sal),
MAX(comm), MAX(deptno)
FROM scott.emp;
```

3.4. O2O CDC (Consistent Set)

以Dept、EMP为例。

3.4.1. 环境准备

为保留之前的练习，也为了避免冲突，建议重新做。创建来源数据库用户Scott2，把Scott下的表和数据都搬过来

```
create user scott2 identified by tiger;
grant connect,resource to scott2;
grant create view to scott2;

-- Create table
create table scott2.DEPT
(
  DEPTNO NUMBER(2) not null,
  DNAME  VARCHAR2(14),
  LOC    VARCHAR2(13)
);
-- Create/Recreate primary, unique and foreign key constraints
alter table scott2.DEPT
  add constraint PK_DEPT primary key (DEPTNO)
  using index;

-- Create table
create table scott2.EMP
(
  EMPNO    NUMBER(4) not null,
  ENAME    VARCHAR2(10),
  JOB      VARCHAR2(9),
  MGR      NUMBER(4),
  HIREDATE DATE,
  SAL      NUMBER(7,2),
  COMM     NUMBER(7,2),
  DEPTNO   NUMBER(2)
);
-- Create/Recreate primary, unique and foreign key constraints
alter table scott2.EMP
  add constraint PK_EMP primary key (EMPNO)
  using index;
alter table scott2.EMP
  add constraint FK_DEPTNO foreign key (DEPTNO)
  references scott2.DEPT (DEPTNO);
INSERT INTO scott2.dept
  SELECT * FROM scott.dept;
INSERT INTO scott2.emp
  SELECT * FROM scott.emp;
```

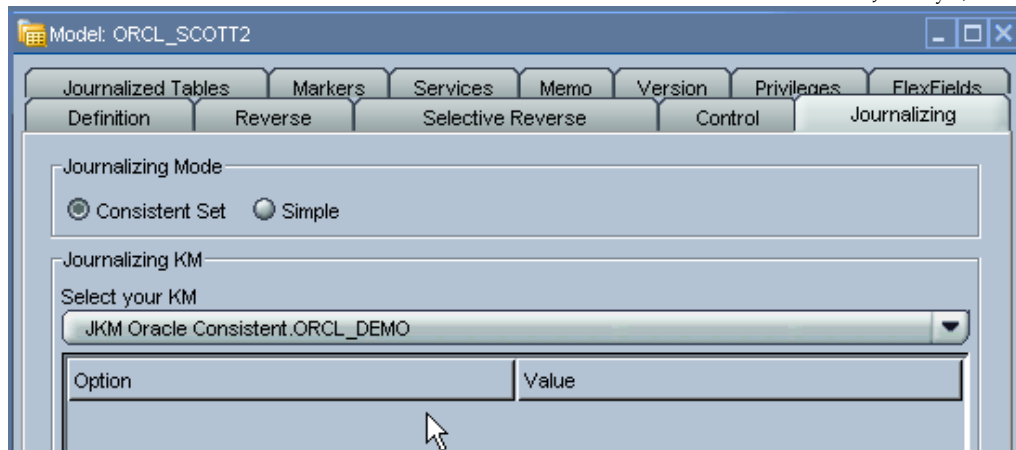
3.4.2. 完成普通设置步骤

参照前面Oracle to Oracle，完成模型创建，比如DEPT、EMP。

3.4.3. 模型日记记录设置

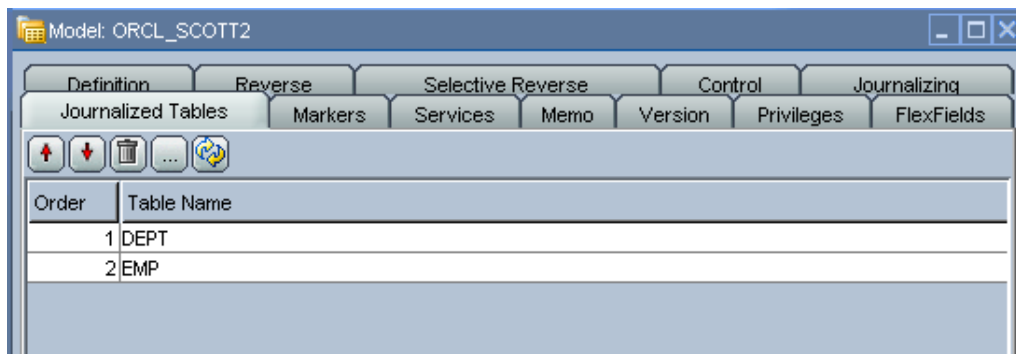
N:Designer\模型\<源模型名，如ORCL_SCOTT2>\双击\日记记录

启用Consistent Set，并选择JKM Oracle Consistent.ORCL_DEMO:



3.4.4. 添加到CDC

N:\Designer\模型\ORCL_SCOTT2\<Datastore> \右键\已更改数据捕获\添加到CDC
把Dept、EMP都添加到CDC，并排列顺序：



另外，Datastore的图标上，左上角多了个黄色的小钟： DEPT。

这个顺序只对Consistent Set的Journalizing有用，当然了，Consistent Set的Journalizing也必须在这里设置顺序，并且如果调整了顺序，需要重启下Journal。

问题：重启Journal，会不会导致数据丢失呢？

3.4.5. 增加订阅

N:\Designer\模型\ORCL_SCOTT2\右键\已更改数据捕获\订户\订阅

增加一个名字叫“DEPT-EMP”的订阅者。

3.4.6. 启动日记

N:\Designer\模型\ORCL_SCOTT2\右键\已更改数据捕获\启动日记

注：增加订阅、增加订阅这两个步骤，也可以在Package中设置，这样方便移植。操作很简单，把Datastore或整个Model拖到Package的Diagram中，Type选择Journalizing Datastore或Model。

也可以针对单个Datastore启动日记。

注：If a datastore is removed from CDC in Consistent Set mode, the journals should be restarted for the model (Journalizing information is preserved for the other datastores).

问题：重启Journal，会不会导致数据丢失呢？

3.4.7. 测试

1、在PL/SQL Developer中执行下列语句往源系统插入数据

```
INSERT INTO scott2.dept
  SELECT MAX(deptno) + 1, 'TEST', 'TEST' FROM scott2.dept;
INSERT INTO scott2.emp
  SELECT MAX(empno) + 1, MAX(ename), MAX(job), MAX(mgr), MAX(hiredate), MAX(sal),
  MAX(comm), MAX(deptno)
  FROM scott2.emp;
```

2、N:<源Data Store如ORCL_SCOTT2下的DEPT、EMP>\右键\已更改数据捕获\日记数据，可以看到，日记中有一条新增记录

3.4.8. 创建普通接口

注：加了CDC功能的Data Store对普通的接口没有任何影响，接口跑完，该过去的的数据都会过去，已更改数据捕获\日记数据中的数据依然存在。

参照前面Oracle to Oracle，接口名为POP.DEPT_O2O_CS，完成接口创建、同步数据。

1、N:<目标Data Store如ORCL_ODITRG下的DEPT>\右键\数据，可以看到所有数据

2、N:<源Data Store如ORCL_SCOTT2下的DEPT>\右键\已更改数据捕获\日记数据，记录还在！

3.4.9. 接口启用CDC

N:Designer\<项目名>\<文件夹名>\接口\POP.DEPT_O2O_CS\双击\关系图

操作和Simple一样，过滤条件是JRN_SUBSCRIBER = 'DEPT-EMP'。

注：JRN_DATE在过滤器中不要用，以免导致数据不一致。

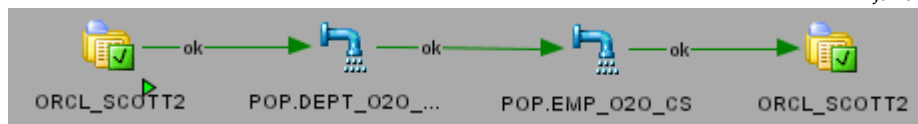
把POP.DEPT_O2O_CS、POP.EMP_O2O_CS都设置下。

运行接口，数据没有任何变化！已更改数据捕获\日记数据中的数据依然存在！这个和Simple的不同了！

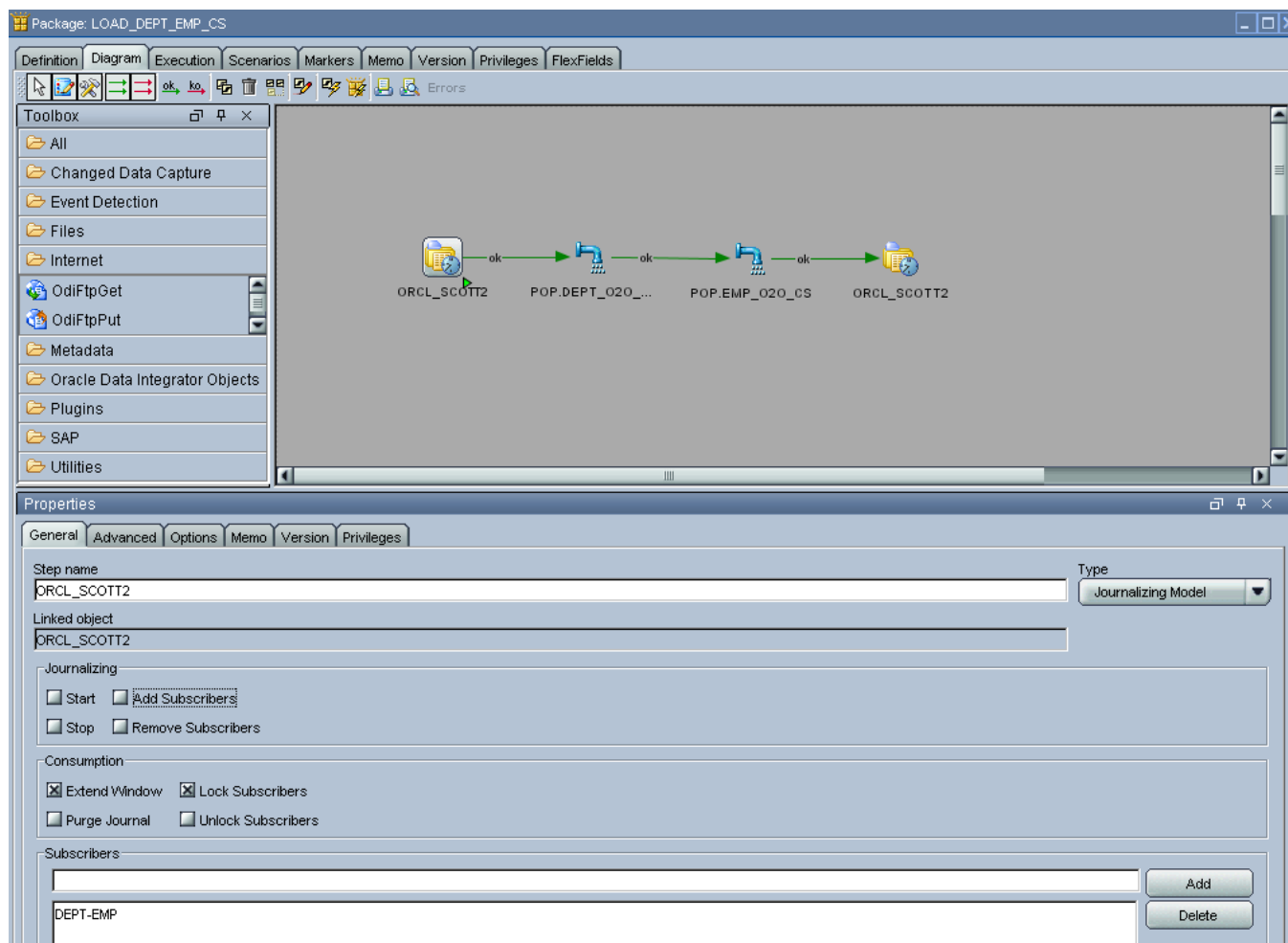
3.4.10. 创建Package

1、创建Package “LOAD_DEPT_EMP_CS”

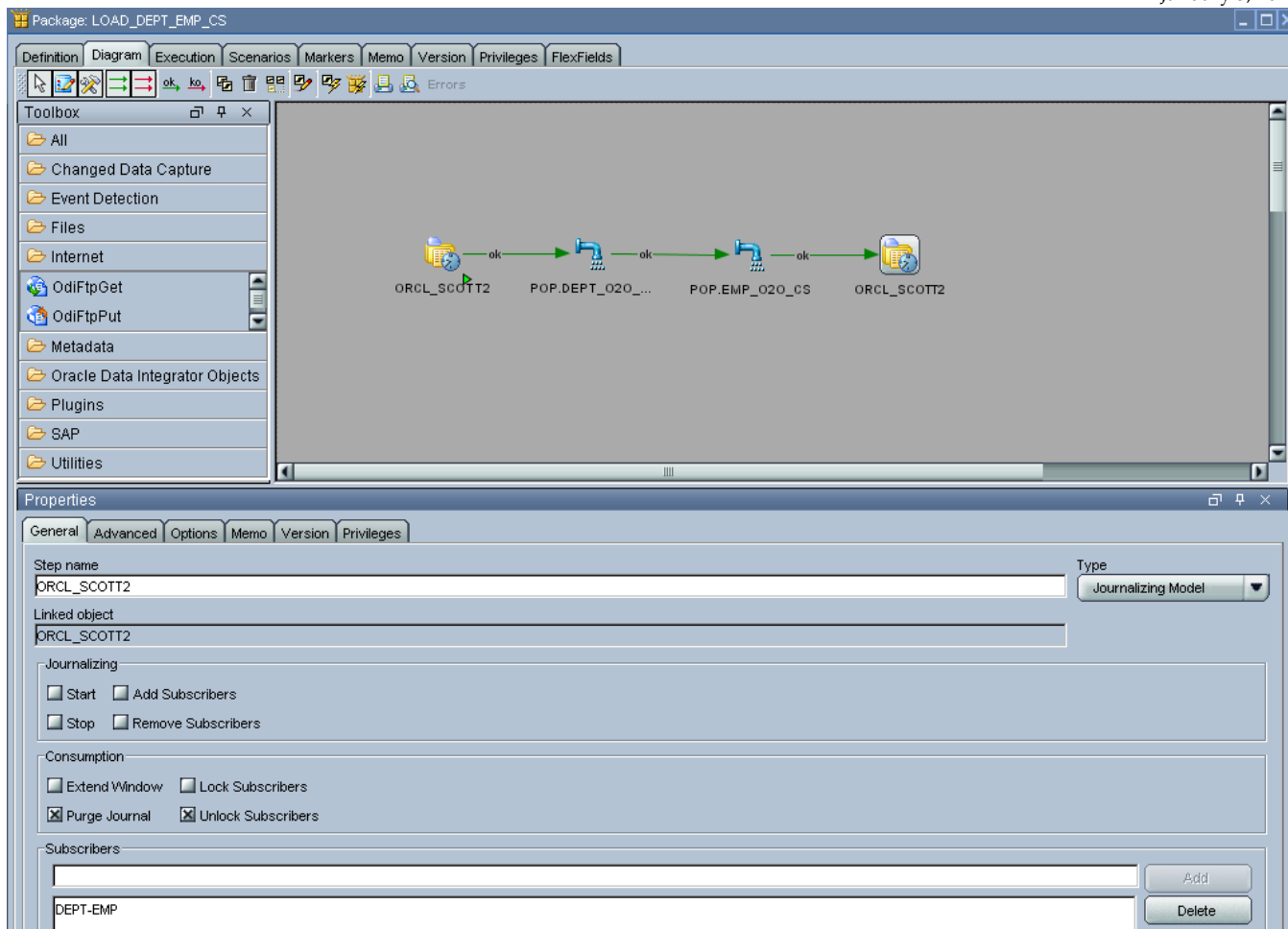
2、把Model “ORCL_SCOTT2” 拖两遍进来，一个做起点，一个做终点，把POP.DEPT_O2O_CS、POP.EMP_O2O_CS也拖进来，设置完前后步骤：



3、设置起点属性，改Type为Journalizing Model，其它根据下图设置



4、设置终点属性，改Type为Journalizing Model，其它根据下图设置



3.4.11. 数据验证

- 1、在PL/SQL Developer中执行下列语句往源系统插入数据

```
INSERT INTO scott2.dept
  SELECT MAX(deptno) + 1, 'TEST', 'TEST' FROM scott2.dept;
INSERT INTO scott2.emp
  SELECT MAX(empno) + 1, MAX(ename), MAX(job), MAX(mgr), MAX(hiredate), MAX(sal),
  MAX(comm), MAX(deptno)
  FROM scott2.emp;
```

- 2、N:<源Data Store如ORCL_SCOTT2下的DEPT、EMP>\右键\已更改数据捕获\日记数据，可以看到，日记中各有一条新增记录
- 3、执行LOAD_DEPT_EMP_CS包
- 4、N:<目标Data Store如ORCL_ODITRG下的DEPT>\右键\数据，可以看到新增数据
- 5、N:<源Data Store如ORCL_SCOTT2下的DEPT>\右键\已更改数据捕获\日记数据，记录不见了

问题：如果有两个Interface，筛选条件为JRN_SUBSCRIBER='DEPT-EMP'，那么第一个Interface跑完，第二个岂不是没数据可抓了？

结论：如果有多个Interface要用到同一个CDC，那么就要定义不同的订阅。

3.5. O2O CDC (Consistent Set Using Log Minner)

以Dept、EMP为例。

3.5.1. 环境准备

- 1、为保留之前的练习，也为了避免冲突，建议重新做。创建来源数据库用户Scott3，把Scott下的表和数都搬过来

```
create user scott3 identified by tiger;
grant connect,resource to scott3;

-- Create table
create table scott3.DEPT
(
  DEPTNO NUMBER(2) not null,
  DNAME  VARCHAR2(14),
  LOC    VARCHAR2(13)
);
-- Create/Recreate primary, unique and foreign key constraints
alter table scott3.DEPT
  add constraint PK_DEPT primary key (DEPTNO)
  using index;

-- Create table
create table scott3.EMP
(
  EMPNO    NUMBER(4) not null,
  ENAME    VARCHAR2(10),
  JOB      VARCHAR2(9),
  MGR      NUMBER(4),
  HIREDATE DATE,
  SAL      NUMBER(7,2),
  COMM     NUMBER(7,2),
  DEPTNO   NUMBER(2)
);
-- Create/Recreate primary, unique and foreign key constraints
alter table scott3.EMP
  add constraint PK_EMP primary key (EMPNO)
  using index;
alter table scott3.EMP
  add constraint FK_DEPTNO foreign key (DEPTNO)
  references scott3.DEPT (DEPTNO);
INSERT INTO scott3.dept
  SELECT * FROM scott.dept;
INSERT INTO scott3.emp
  SELECT * FROM scott.emp;
```

- 2、创建Work Schema用的用户，最重要的是授权

```
create user SOA_USER identified by SOA_USER;
grant connect, resource to SOA_USER;
grant create view to SOA_USER;
grant create database link to SOA_USER;
grant create synonym to SOA_USER;
grant select any table to SOA_USER;
Grant EXECUTE_CATALOG_ROLE , SELECT_CATALOG_ROLE to soa_user;
GRANT EXECUTE ON DBMS_CDC_PUBLISH TO soa_user;
```

3.5.2. 与普通Consistent Set的不同

在使用ODI上，仅仅是JKM不同，这里选JKM Oracle 10g Consistent (LOGMINER)

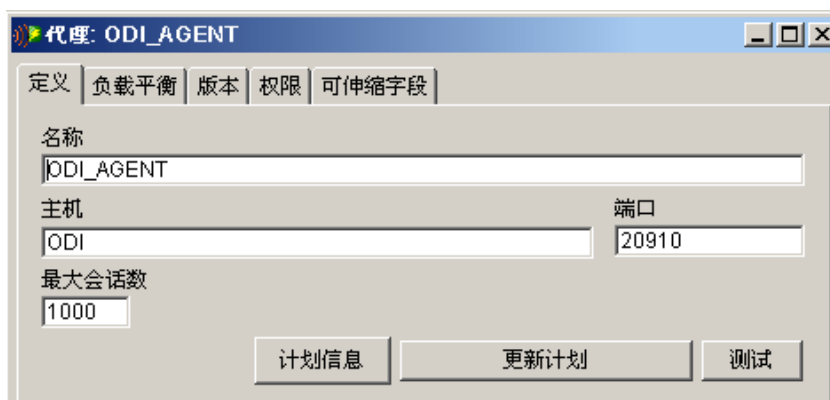
3.6. Agent

3.6.1. 启动Agent

N:开始菜单\Agent

3.6.2. 创建物理Agent

N:Topology Manager\物理体系结构\代理\右键\插入代理



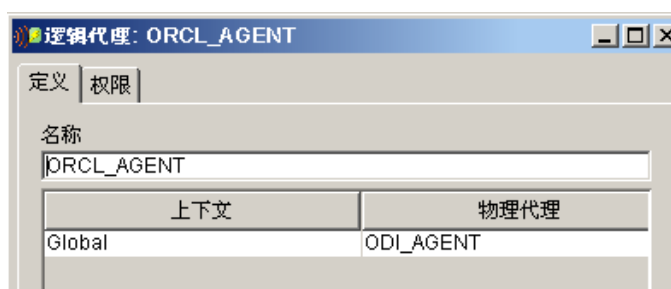
主机：填写启动Agent的主机名或IP地址，这里是本机机器名ODI

端口：默认是20910

点击“测试”看是否能够连通。

3.6.3. 创建逻辑Agent

N:Topology Manager\逻辑体系结构\代理\右键\插入代理



3.6.4. 用Agent运行

N:Topology Manager\逻辑体系结构\代理\右键\插入代理

当运行某个Interface时，除了“本地(无代理)”外，就可以选择“ORCL_AGENT”了。

3.7. Schedule

3.7.1. 创建方案计划

N:某个方案（Scenario）\计划\右键\插入计划

一个方案可以有多个执行计划，一个执行计划只属于一个特定的方案。

1、定义上下文、代理、日志级别、活动期间、执行频率

方案计划: GLOBAL

定义 执行循环 Variables 权限 版本

方案: DEPT / 001

上下文: Global 代理: ORCL_AGENT

日志级别: 5

活动: ☒ 活动 ☐ 非活动 ☐ 活动期间:

☐ 开始: the 2009-3-11 at 14:26:23

☐ 结束: the 2009-3-11 at 14:26:23

☐ 在以下日期范围之间每天: 从 14:26:23 到 14:26:23

☐ 除了一个月中的这些天:

☐ 除了一个星期中的这些天: ☐ 星期一 ☐ 星期二 ☐ 星期三 ☐ 星期四 ☐ 星期五 ☐ 星期六 ☐ 星期日

执行: ☒ 启动时 ☐ 简单 ☐ 每小时 ☐ 每天 ☐ 每周 ☐ Monthly (day of the month) ☐ 每月(工作日) ☐ 每年

确定 取消 应用 帮助

2、定义每次执行时的循环次数

方案计划: GLOBAL

定义 执行循环 Variables 权限 版本

重复: ☐ 无(执行一次) ☒ 多次

☒ 最大重复数: 20

☐ 最大循环持续时间: <未定义>

重复间隔: 5 秒

约束: ☐ 失败时的尝试次数: <未定义> ☐ 停止执行条件: <未定义>

确定 取消 应用 帮助

3.7.2. 编辑配置文件

先用agent编码两个密码:

```
cd C:\OraHome_1\oracledi\bin
agent encode snpm
agent encode SUNOPSIS
```

编辑C:\OraHome_1\oracledi\bin\odiparams.bat, 注意修改以下内容:

```
set ODI_SECU_DRIVER=oracle.jdbc.driver.OracleDriver
set ODI_SECU_URL=jdbc:oracle:thin:@HUAJHUA:1522:ORCL
set ODI_SECU_USER=snpm
set ODI_SECU_ENCODED_PASS=d,yHO9LLt84x1Ya8brTZ,ATCe
set ODI_SECU_WORK_REP=WORKREP1
set ODI_USER=SUPERVISOR
set ODI_ENCODED_PASS=fDyXNs.YMr8A66ddjLIy
```

其中两个_ENCODED_PASS就是上面编码过的密码串。

3.7.3. 启动Scheduler Agent

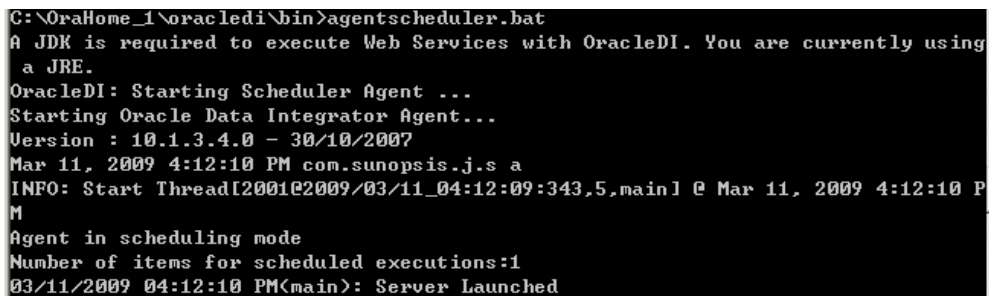
先停掉原有的Agent, 不然报“java.net.BindException:Address already in use: JVM_Bind”。

启动命令如下:

```
cd C:\OraHome_1\oracledi\bin
agentscheduler.bat
```

注: 如果报错, 可以加上port、name、v等参数。agentscheduler ["-PORT=<port>"] ["-NAME=<agent name>"] ["-V=<trace level>"]

正确的话, 就可以看到如下结果:



```
C:\OraHome_1\oracledi\bin>agentscheduler.bat
A JDK is required to execute Web Services with OracleDI. You are currently using
a JRE.
OracleDI: Starting Scheduler Agent ...
Starting Oracle Data Integrator Agent...
Version : 10.1.3.4.0 - 30/10/2007
Mar 11, 2009 4:12:10 PM com.sunopsis.js a
INFO: Start Thread[200102009/03/11_04:12:09:343,5,main] @ Mar 11, 2009 4:12:10 P
M
Agent in scheduling mode
Number of items for scheduled executions:1
03/11/2009 04:12:10 PM(main): Server Launched
```

然后到Operator那里, 可以看到完成的各个会话。

4. 最常用特性和功能实例二（SQL Server、XML）

4.1. Oracle to SQL Server

以Oracle 10.2到SQL Server 2000为例。

4.1.1. SQL Server环境准备

- 1、下载并安装SQL Server 2000，认证采用混合验证模式。
- 2、下载并安装SQL Server 2000SP4，SP4不装的话，TCP/IP的端口没有打开，无法通过JDBC访问。
- 3、创建EMP表

```
CREATE TABLE [dbo].[EMP] (  
    [EMPNO] [int] NOT NULL ,  
    [ENAME] [char] (10) COLLATE Chinese_PRC_CI_AS NULL ,  
    [JOB] [char] (9) COLLATE Chinese_PRC_CI_AS NULL ,  
    [MGR] [int] NULL ,  
    [HIREDATE] [datetime] NULL ,  
    [SAL] [numeric](7, 2) NULL ,  
    [COMM] [numeric](7, 2) NULL ,  
    [DEPTNO] [int] NULL  
) ON [PRIMARY]  
GO
```

```
ALTER TABLE [dbo].[EMP] ADD  
    CONSTRAINT [IX_EMP] UNIQUE NONCLUSTERED  
    (  
        [EMPNO]  
    ) ON [PRIMARY]  
GO
```

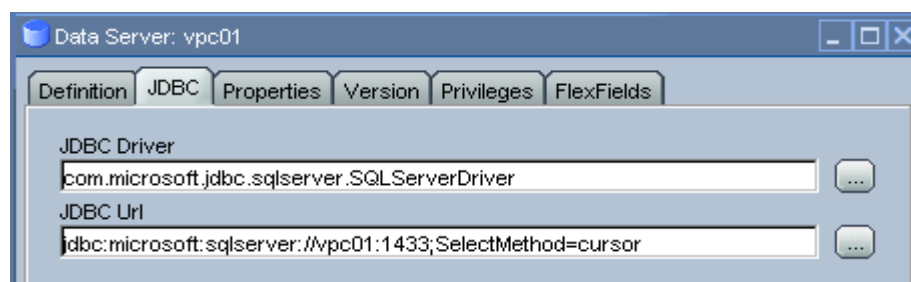
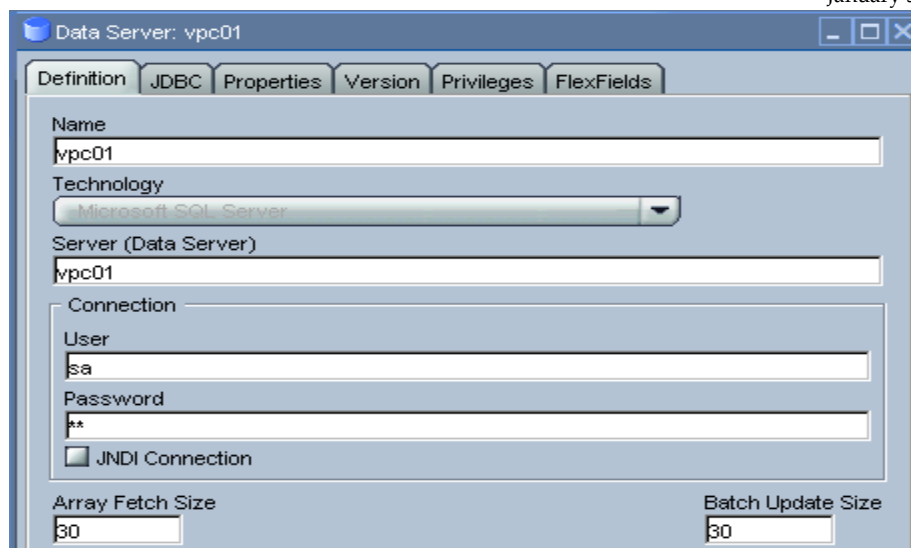
4.1.2. ODI环境准备

下载并安装mssql drive for jdbc.exe，然后把C:\Program Files\Microsoft SQL Server 2000 Driver for JDBC\lib下面的3个jar文件拷贝到ODIHome\oracledi\drivers下。

注：本文档附件“mssql drive for jdbc”文件夹下包含了上述3个文件，可直接使用。

4.1.3. ODI设计要点

- 1、Data Server（假定数据库服务器在VPC01上，用SA连接）



JDBC Driver: `com.microsoft.jdbc.sqlserver.SQLServerDriver`

JDBC URL: `jdbc:microsoft:sqlserver://vpc01:1433;SelectMethod=cursor`

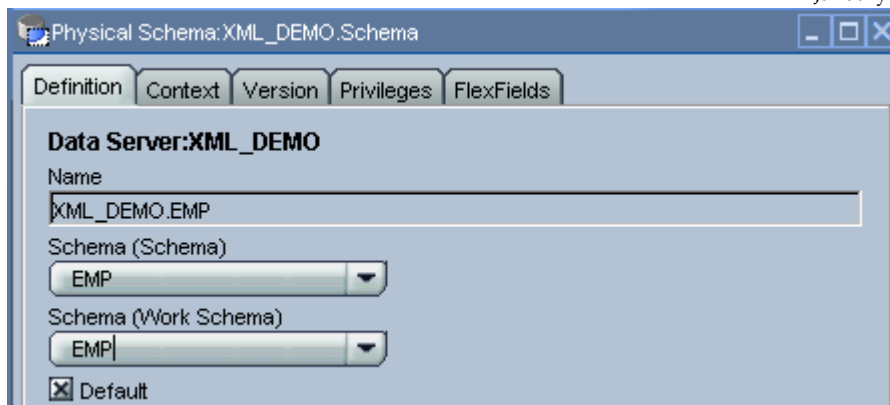
2、Physical Schema



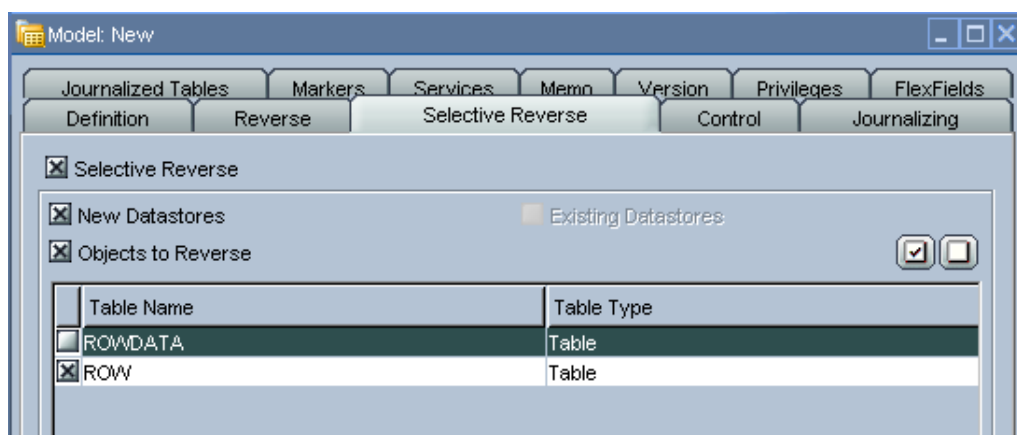
3、其他步骤与Oracle to Oracle类似，也要目标Module先创建主键，不同点就是自动选择的各个KM不一样。

4.2. SQL Server to Oracle

没有什么特殊的。



- 3、创建模型，会看到有两个表，这是因为，Oracle Data Integrator Driver for XML，在把层次性的XML文件Mapping成关系型数据时，非底层Element翻译成表，底层Element和Attribute翻译成列。如果Element带属性，那么该Element即使是底层的，也会被翻译成子表？



这里我们只选ROW。

- 4、在ODI数据模型中View Data，可以看到有两类特殊字段：<xx>ORDER、<上层表>FK。参阅Oracle Data Integrator Driver for XML - User's Manual。

Data ROW

Data

</

- 5、其他步骤与Oracle to Oracle类似，也要目标Module先创建主键，不同点就是自动选择的各个KM不一样。

4.3.3. XML数据不同步问题

Oracle Data Integrator Driver for XML在将xml文件数据load到内存后，要使两者的数据同步，需要：

- 1、重新连接，比如在设计阶段，退出Designer再登录，View Data才能看到xml文件的变化。这个肯定是不行的。
- 2、执行特定命令，SYNCHRONIZE FROM FILE用来把文件同步到内存，SYNCHRONIZE ALL或 SYNCHRONIZE FROM DATABASE用把内存同步到文件。这个需要在Package中调用，也就是在Interface的前面，调用SYNCHRONIZE FROM FILE，在Interface的后面调用SYNCHRONIZE ALL。

以上可通过修改知识模块实现，请参见本文档“客户化KM最佳简单例子”。也可以通过创建Procedure然后在Package中调用实现，请参见本文档“Procedure最简单的例子”。

- 3、在创建Data Server时，JDBC URL添加两个参数&ro=true&dod=yes，适合XML为只读数据源的情况：`jdbc:snps:xml?f=../demo/xml/emp.xml&ro=true&dod=yes`

注：OracleDI Driver for XML除了用内存，还可以用普通的关系型数据库来存储转换后的xml数据。详见Oracle Data Integrator Driver for XML - User's Manual

4.3.4. XML根元素改变问题

如果XML根元素，ODI将报错“Root element was not found in the XML Model”。这个目前尚无解决方法。实际上，也比较好理解，XML元素就相当于表名，如果改变了，ODI当然“认不出来了”。

5. Sequence、Variable、User Function、Procedure

5.1. 作用域

ODI中Sequence、Variable、Function、Marker，都有两种作用域Project、Global（在Others标签页），并且Project优先于Global。

5.2. Variable变量

5.2.1. 关于变量

ODI中的变量，属于高级应用，用于参数化ODI的各个对象，使得我们设计出来的ETL具有极高的灵活性。主要在如下几个场合使用：

- 1、各个地方的表达式（比如铅笔按钮点开的界面）和SQL
- 2、图形模块，如Datastore界面的Resource栏位，动态决定表名；再如Physical Schema界面的Schema (Schema)，动态决定数据所在的Schema
- 3、通过OS命令执行Scenario时的参数中，也支持变量的传入
- 4、Package中可以：
 - a) 声明变量Declare Variable，看起来没用，但Oracle强烈推荐在用前显示声明
 - b) 刷新变量Refresh Variable，当变量来源于SQL时特别有用
 - c) 设置变量Set Variable，用表达式设置，数值型变量还可设置增量
 - d) 比较变量Evaluate Variable，用于决定流程分支时特别有用

变量的引用有两种：

- 1、替换法：如#<Variable>、#GLOBAL.<Variable>、#<Project>.<Variable>
- 2、邦定法：如:<Variable>、:GLOBAL.<Variable>、:<Project>.<Variable>，但可能影响执行计划，所以不推荐使用

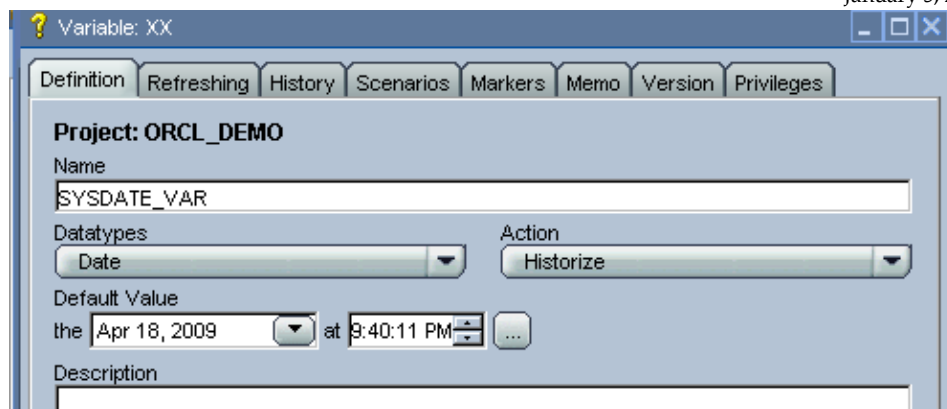
注：由Agent或图形模块在运行时把Variable替换成实际的值。

以上只是简单说明，请一定要参考User Guide中的Creating and Using Variables。

5.2.2. 变量举例

以SQL变量系统日期为例：

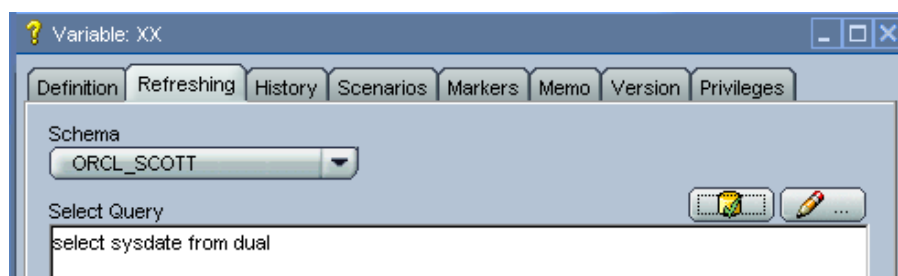
- 1、定义变量SYSDATE_VAR，注意名字区分大小写



数据类型中，字符长度为255，数字为10，Text长度无限制。

Action指ODI保留变量值的时间范围，Historize每个历史值都保存在资料库中，Last Value只保存最后一个值在资料库中，Non Persistent保存在会话内存中。可在History标签页查看变量值。

2、在Refreshing标签页定义SQL



- 3、刷新变量，点击按钮“Refresh”对变量进行刷新，也可在Package中调用
- 4、查看变量，在History标签页，点击刷新图标，可以查看变量历史值
- 5、创建Scenario，与Interface、Procedure、Package一样，变量也可创建Scenario

5.3. Sequence序列

5.3.1. 关于序列

ODI中的序列是特殊的变量，每次用完自动增长。有两种：一种是存在资料库中的标准序列，另一种是用外部表的特殊序列，都由ODI自动管理，比如通过锁确保并发读取时的唯一性。序列的范围有两种：Project、Global（在Others标签页）。

如果RDBMS有序列功能，如Oracle和SQL Server，就不要用使用ODI的序列。

序列可在各个地方的表达式（比如铅笔按钮点开的界面）和SQL中引用，并由Agent在运行前替换成实际的值。

序列的引用有两种：

- 1、替换法：如#<Sequence>_NEXTVAL、#GLOBAL.<Sequence>_NEXTVAL、#<Project>.<Sequence>_NEXTVAL

- 2、邦定法：如:<Sequence>_NEXTVAL、:GLOBAL.<Sequence>_NEXTVAL、:<Project>.<Sequence>_NEXTVAL，用于SQL变量邦定

注：由Agent或图形模块在运行时把Sequence替换成实际的值，insert into <t1>(...) select #<Sequence>_NEXTVAL,... from <t2>，因为Agent只执行一次，所以不管该Insert有多少条记录，序列都是一样的。

To make sure that a sequence is updated for each row inserted into a table, each row must be processed by the Agent. To make this happen, follow the steps below:

- 1、 Make the mapping containing the sequence be executed on the target.
- 2、 Set the mapping to be active for inserts only. Updates are not supported for sequences.
- 3、 If you are using an "incremental update" IKM, you should make sure that the update key in use does not contain a column populated with the sequence. For example, if the sequence is used to load the primary key for a datastore, you should use an alternate key as the update key for the interface.
- 4、 If using Oracle Data Integrator sequences with bind syntax (:<SEQUENCE_NAME>_NEXTVAL), you must configure the data flow such that the IKM transfers all the data through the agent. You can verify this by checking the generated integration step in Operator . It should have separate INSERT and SELECT commands executed on different connections, rather than a single SELECT...INSERT statement.

Note: Please note the following limitations:

- A column mapped with a sequence should not be checked for not null.
- Similarly, static control and flow control are cannot be performed on a primary or alternate key that references the sequence.

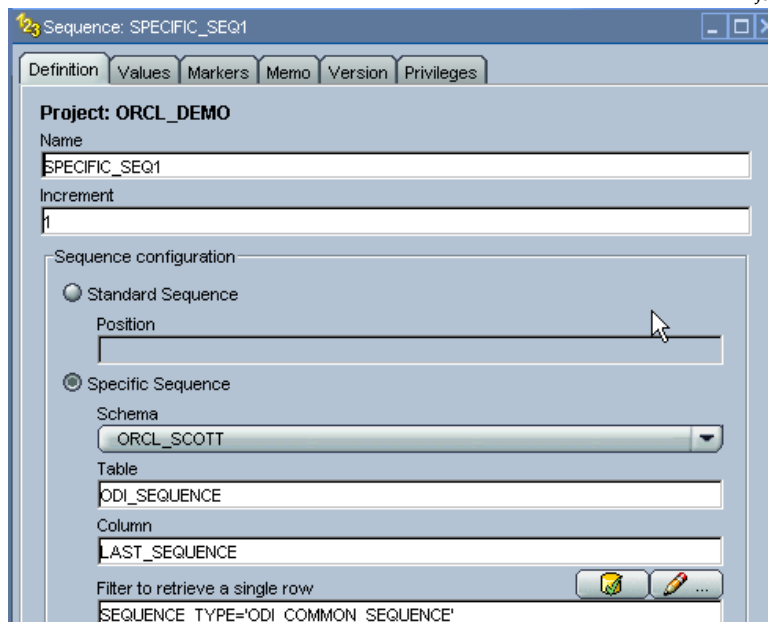
以上只是简单说明，请一定要参考User Guide中的Creating and Using Sequences

5.3.2. 序列举例

标准序列比较简单，下面举特殊序列的例子。先准备外部序列表和数据：

```
-- Create table
create table ODI_SEQUENCE
(
    SEQUENCE_TYPE VARCHAR2(30) not null,
    LAST_SEQUENCE NUMBER not null
);
-- Add comments to the columns
comment on column ODI_SEQUENCE.SEQUENCE_TYPE
    is '序列类型';
comment on column ODI_SEQUENCE.LAST_SEQUENCE
    is '最后一个序列';
-- Create/Recreate indexes
create unique index ODI_SEQUENCE_U1 on ODI_SEQUENCE (SEQUENCE_TYPE);
-- Insert
insert into ODI_SEQUENCE values ('ODI_COMMON_SEQUENCE',1)
```

创建特殊序列SPECIFIC_SEQ1，假定上面的表就创建在ORCL_SCOTT这个数据源中



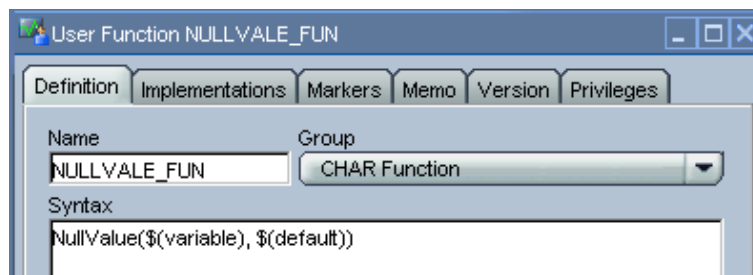
5.4. User Function自动义函数

5.4.1. 关于自定义函数

不同的技术，比如Oracle和SQL Server，其支持的函数各不相同，有些即使功能相同，语法比如函数名也不同。自定义函数，可提供一个统一的、简单的函数，然后对应到不同技术的不同函数。当然也可以有其它自定义用途。可在Interface和Procedure中调用。总的来说，这中做法应该不常用。

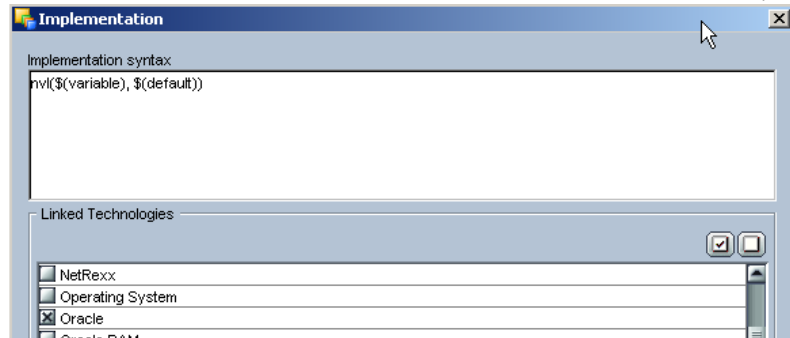
5.4.2. 自定义函数举例

1、定义函数NULLVALE_FUN及其语法



NullValue\$(variable), \$(default))

2、在Implementations标签页定义函数在不同技术下的实现



Oracle: `nvl$(variable), $(default))`

SQL Server: `case when $(variable) is null then $(default) else $(variable) end`

5.5. Procedure过程

5.5.1. 关于Procedure

Procedure也用来执行一系列命令，不过是顺序执行，没有分支的概念，只有一个选项“Ignore Errors”控制当出错时是停止执行还是继续执行下一个命令。此外，也可以通过参数控制一个命令是否需要执行，比如是否执行创建索引、是否执行创建DBLINK等。

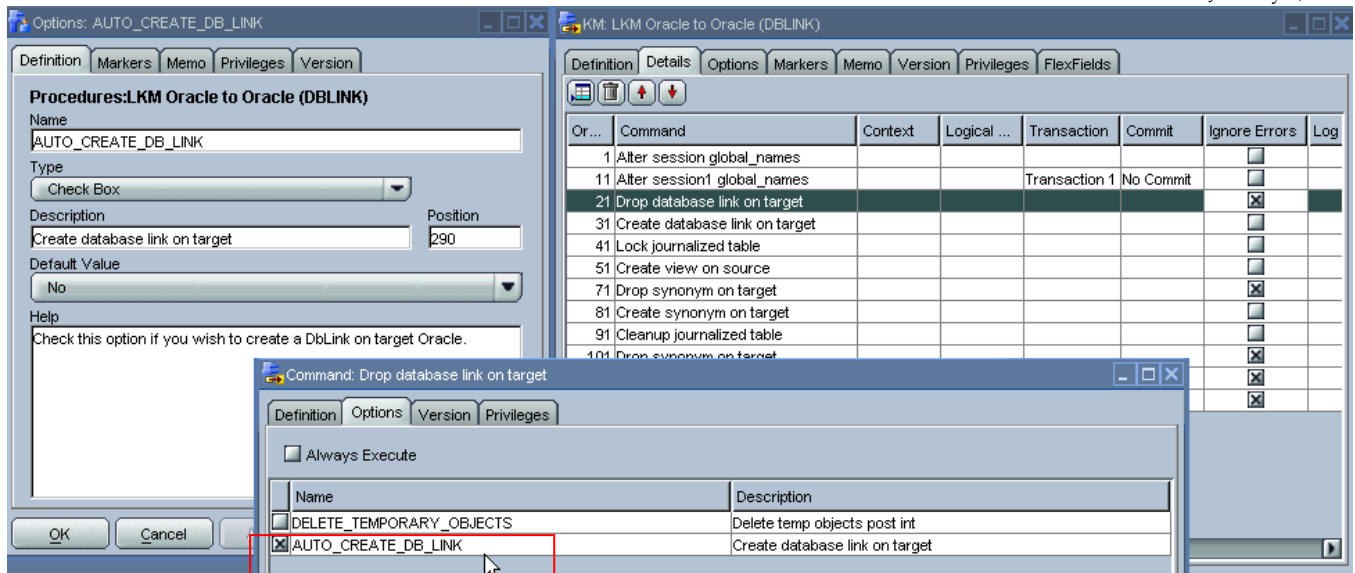
实际上，所有的知识模块，本质上也是Procedure。所以最直观的理解，就是双击某个KM，看Detail标签，就知道Procedure能干什么了。它可以：

- 1、执行ODI命令
- 2、执行SQL语句
- 3、执行数据库存储过程
- 4、.....

5.5.2. Option

右键Procedure，可以插入Option，类型有Check Box、Value和文本，这些Option设计时可以让用户设置，运行时通过ODI的API获取到，也就是这里也提供了一个参数化的功能。此外，Check Box类型的Option，还可以直接用来控制某个命令是否执行。

LKM Oracle to Oracle (DBLINK)有个Option叫AUTO_CREATE_DB_LINK，默认是No，控制Drop database link on target是否执行，如下图红框所示



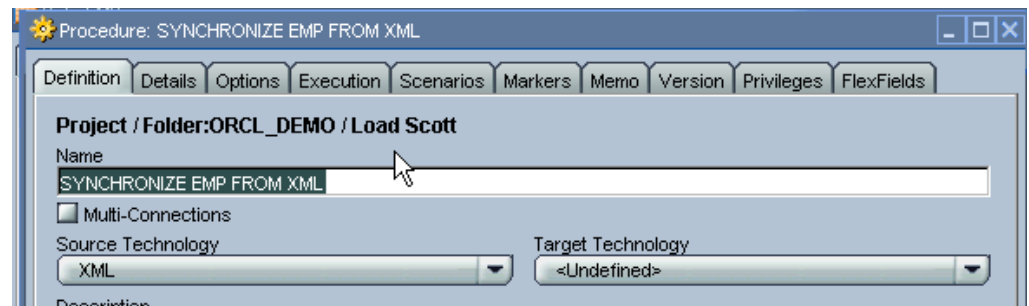
此外，所有Option，可在运行前，在Procedure的Option标签页统一设置。

5.6. Procedure最简单的例子

以从XML文件同步的命令，解决XML文件内容变化而ODI不知道的问题。

5.6.1. 创建Procedure

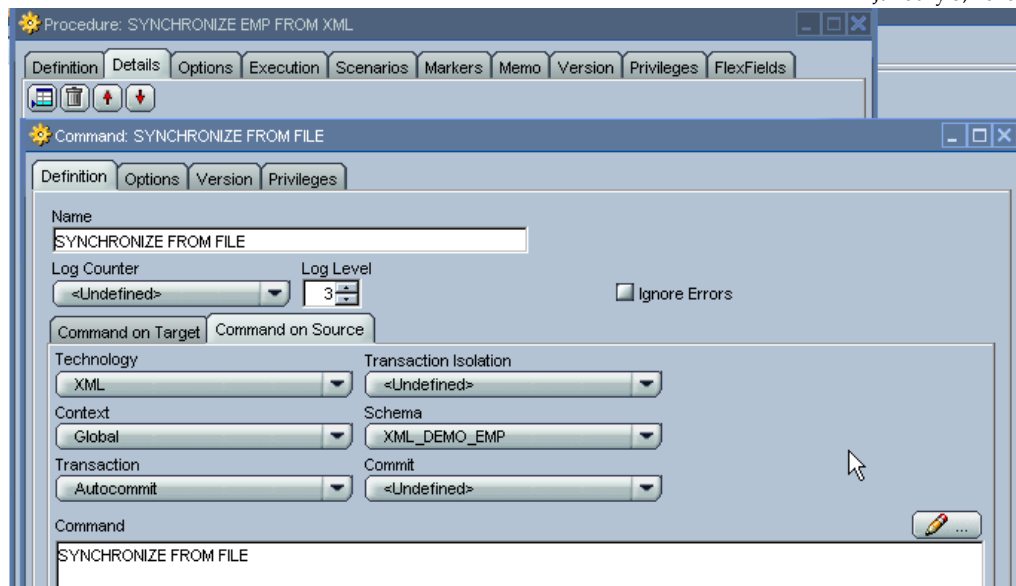
N:Designer\<项目名>\<项目>\<文件夹>\Procedures\右键\插入Procedure



名字为SYNCHRONIZE EMP FROM XML，源技术为XML。

5.6.2. 增加命令

在Details标签页，新增命令SYNCHRONIZE FROM FILE:



注：命令写在Source标签页，并且需要选择Shcema。

5.6.3. 运行Procedure

运行方式与Interface类似，点击Excute按钮或右键Procedure。然后就可以到Operator中查看运行情况。

5.7. 在Procedure中用序列、函数

5.7.1. 准备数环境

脚本如下：

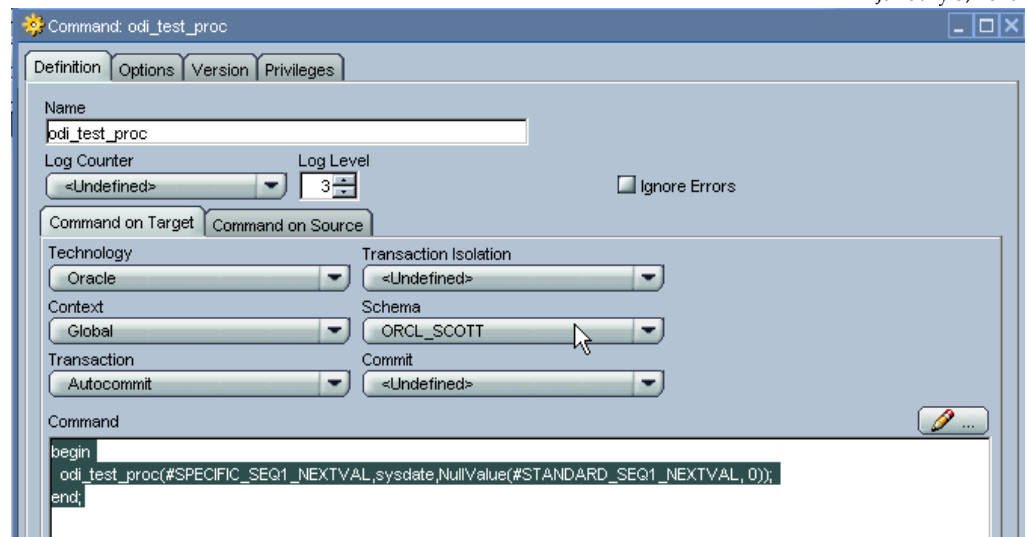
```
-- Create table
create table ODI_PROC_TEST
(
  P1 NUMBER,
  P2 DATE,
  P3 NUMBER
);

-- Create procedure
CREATE OR REPLACE PROCEDURE odi_test_proc(p1 IN NUMBER, p2 IN DATE, p3 IN NUMBER) IS
BEGIN
  INSERT INTO odi_proc_test VALUES (p1, p2, p3);
END;
```

5.7.2. 创建ODI Procedure

创建过程TEST_PROC，步骤略，命令如下：

```
begin
  odi_test_proc(#SPECIFIC_SEQ1_NEXTVAL,sysdate,NullValue(#STANDARD_SEQ1_NEXTVAL, 0));
end;
```



说明：SPECIFIC_SEQ1是上面例子创建的基于SQL的序列，STANDARD_SEQ1则是我另外创建的普通序列，NullValue是上面例子创建的自定义函数，odi_test_proc是数据库中的一个过程。

5.7.3. 测试

在ODI中Execute Procedure，如果没有错，到PL/SQL Developer中执行 `select * from odi_proc_test` 将看到数据插进来了。

6. Knowledge Module

6.1. 客户化KM最佳简单例子

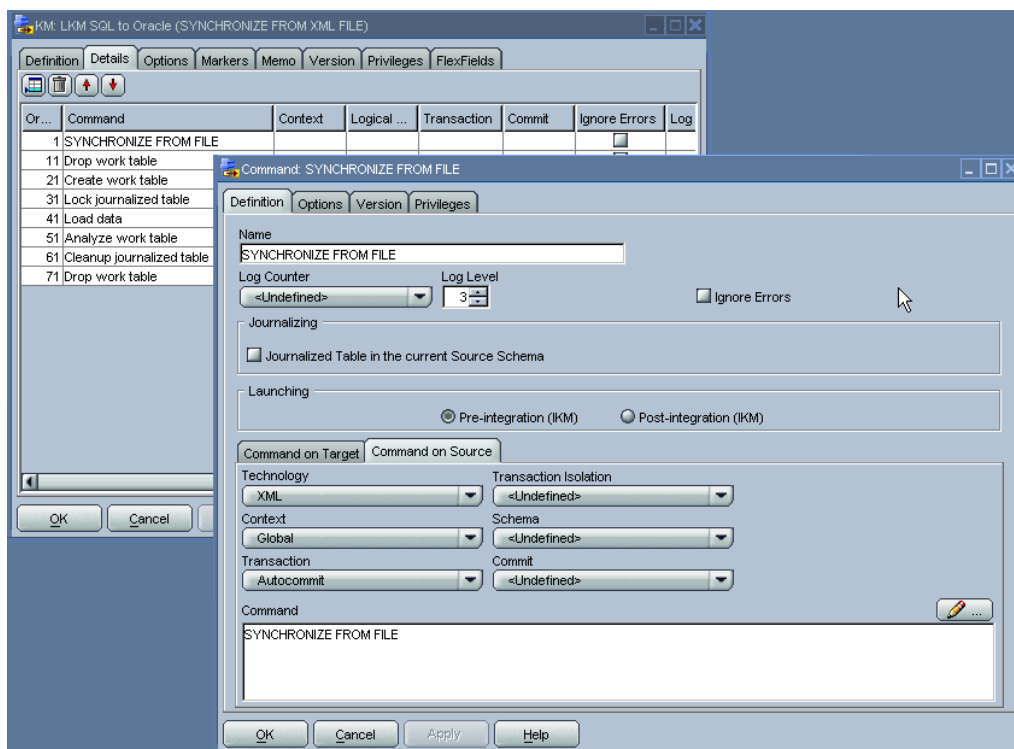
这里以修改LKM SQL to Oracle为例，添加从XML文件同步的命令，解决XML文件内容变化而ODI不知道的问题。

6.1.1. Copy已有KM

拷贝LKM SQL to Oracle，然后该名字为LKM SQL to Oracle (SYNCHRONIZE FROM XML FILE)。

6.1.2. 添加

在Detail标签页，添加一个SYNCHRONIZE FROM FILE命令，并将它移到第一步：



注意，命令写在Command on Source标签页。这里不需要指定Schema，运行时自动用当前的Schema。

7. Package及ODI工具箱

7.1. 概述

7.1.1. Package的作用

Package是ODI中最大的运行单元，运行一系列有前后关联的任务，它将以下对象组成一个可运行的Workflow：

- 1、 Interface
- 2、 Variable
- 3、 Procedure
- 4、 ODI Tools
- 5、 Model
- 6、 Datastore

8. Web Service

8.1. 安装Public Web Services

以Tomcat 5.5.26为例。

8.1.1. 下载AXIS2

Metalink Notes 550215.1说，与10.1.3.X兼容性好的是1.2版本，其他版本，高了低了都有问题。

登录http://ws.apache.org/axis2/download/1_2/download.cgi下载WAR包。

8.1.2. 下载AXIS2

8.2. 设置Data Services

以Tomcat 5.5.26为例。

9. FAQ&How To

9.1. 常见问题

9.1.1. 界面语言改为英文

修改odiparams.bat:

```
set ODI_ADDITIONAL_JAVA_OPTIONS="-Duser.language=en" "-Duser.region=US".
```

9.1.2. 在delete previous check sum点出错

在delete previous check sum点出错。

“说明” 标签页代码:

```
delete from   Exception getObjectNamesDefaultPSchema("L", "SNP_CHECK_TAB",
"W") :
where        SCHEMA_NAME    = 'ODITRG'
and ORIGIN    = '(6001)ORCL_DEMO.POP.EMP'
and ERR_TYPE  = 'F'
```

出错信息:

```
933 : 42000 : java.sql.SQLException:ORA-00933: SQL 命令未正确结束
java.sql.SQLException:ORA-00933: SQL 命令未正确结束
    at oracle.jdbc.driver.DatabaseError.throwSQLException(DatabaseError.java:125)
    at oracle.jdbc.driver.T4CTTIoer.processError(T4CTTIoer.java:316)
    .....

```

原因: 物理架构的“默认”没有勾选。

9.2. 中英文名词

ODI的文档基本都是英文的, 而Designer等界面默认是中文, Oracle翻译的不好, 很难对照。

Repository	资料库
Model	模型
DataStore	数据存储, 通常就是一个表
Controls	控制
Flow	流
Scenario	方案
Schema	方案
Interface	接口
Package	包
Diagram	关系图

9.3. 对象加密

9.3.1. 对象加密

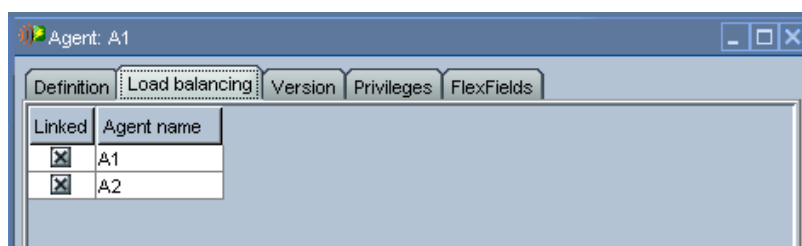
Procedure、KM、Scenario都可以采用DES加密，如果知道Key，也可以解密。详细操作请参考User Guide。

9.4. Agent负载均衡

9.4.1. Agent的LB

在物理Agent上可以设置最大的Session数，默认是1000，在Load Balancing标签页，可以设置链接的其他Agent。

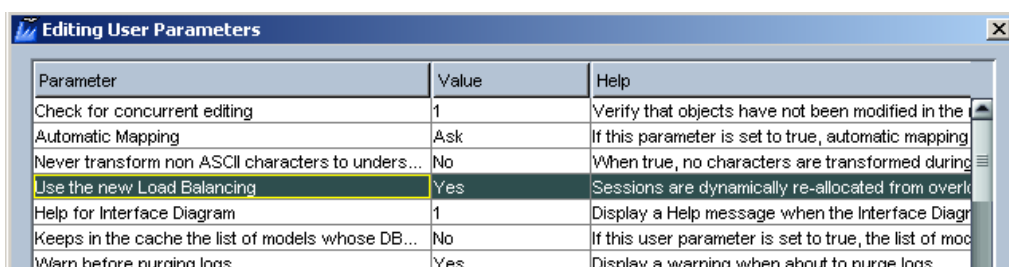
如果不链接自身，那么自身这个Agent将只是用来分发，不具体执行。



Agent的LB原理是根据每个Agent的“正在运行的Session数 / 最大Session数”，来分配新进来的请求要在哪个Agent上跑。

9.4.2. 动态LB

N:File/User Parameters中有个参数叫Use new load balancing，用来控制是否重新分配会话。



9.5. Jython

9.5.1. Jython

ODI的脚本语言是Jython（Python的Java版本），ODI的Agent包含了Jython的解析器，所以可以运行用Jython编写的脚本。在ODI中Jython可以与SQL、PL/SQL、操作系统命令混合编写。

Jython可以在Procedure和KM中（技术设置为Jython即可）。It is possible to perform complex processing with strings, lists, "dictionaries", call FTP modules, manage files, integrate external Java classes, etc.

参考Jython Quick Reference。

9.5.2. 应用

1、非25端口SMTP、SMTP需要验证、需要指定Mail字符集

用OdiSendMail是无法实现的，这时可通过Jython编写Procedure来解决。

Doc ID: 424304.1、424328.1、423953.1

2、

9.6. Substitution Methods

参考Substitution Methods Reference。

9.7. 常用代码块

这里记录一些常用的Jython代码和odiRef方法。

9.7.1. CASEWHEN

类似PL/SQL Decode的写法：

```
CASEWHEN(SRC_CUSTOMER.DEAR=0,'MR',CASEWHEN(SRC_CUSTOMER.DEAR=1,'MRS','MS'))
```

9.8. 命令行工具

9.8.1. 执行方案（Scenario）

```
cd C:\OraHome_1\oracledi\bin
```

```
startscen LOAD_SALES_ADMINISTRATION 001 GLOBAL "-v=2"
```

9.8.2. 启动代理（Agent）

```
cd C:\OraHome_1\oracledi\bin
```

```
agent -help  
agent
```

如果不指定端口号，默认就20910，是类似PL/SQL Decode的写法：

9.8.3.

```
cd C:\OraHome_1\oracledi\bin
agentscheduler.bat -help
agentscheduler.bat
```

9.9. 升级

9.9.1. 10.1.3.4升级到10.1.3.5

假定资料库放在Oracle Database 10GR2。

- 1、 安装10.1.3.5到新Home
- 2、 重新配置下主资料库连接、工作资料库连接
- 3、 在开始菜单中找到Master Repository Upgrade，选择上面创建的主资料库连接，进行升级
- 4、 进入Topology Manager，找到工作资料库，如果有多个，一一右键/Upgrade

9.10. 用户权限

9.10.1. 概述

ODI用Security Manager来管理用户的权限，本质是通过类似OO的方法，把对象、方法授权给用户。

权限有2级：Object及其Method；Object的Instance及其Method。

所有的ODI元素，项目、模型、接口、Datastore、列、约束、会话、方案等等都是对象。

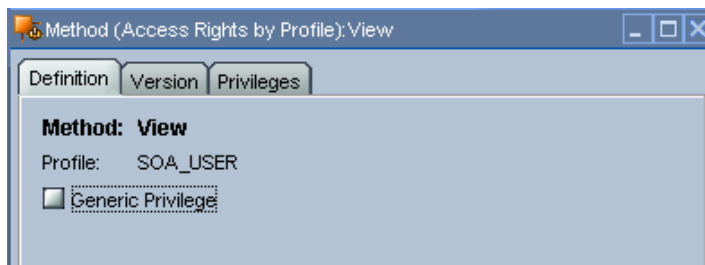
对象的方法，不同对象略有差异，基本有View、Edit、插入、删除等等。

授权有2种：先创建Profile，再把Profile授权给用户；直接把Object及其Method授权给用户。ODI预制了几个常用的Profile，其中Connect是登录ODI各工具的最低权限。

Profile实际上是一组对象，所以下面只说Object，道理一样。

权限管理策略有两种：

- 1、 不分Instance的简单方法，即Object授权，对所有Object的Instance有效。这个通过勾选方法的Generic Privilege来实现。
- 2、 区分Instance的复杂方法，即授予对象权限，但不勾选方法的Generic Privilege，具体有哪些实例的权限，需要在Intance下具体指定。



Object的Instance及其Method，只能直接授权给用户（打开Designer，把对象拖到Instances下），没法通过Profile。

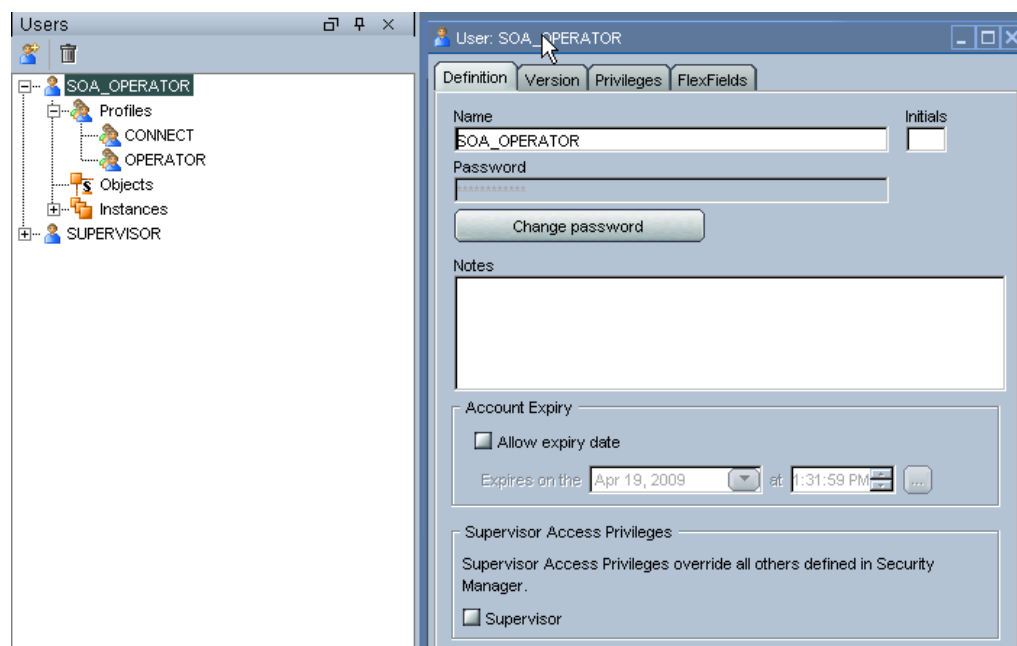
注：因为对象存在关联和引用，我们很难通过授权单个对象给用户而完成授权任务，必须把关联的对象适当权限（如View）全部授权给用户，才能达到预期效果。

9.10.2. 对象关系图

这里提供一个较完整的关系图，在授权时可以用来参考一个对象关联了哪些对象。

9.10.3. 例子：Operator权限

- 1、登录Security Manager，创建用户SOA_OPERATOR，密码和用户名一样。
- 2、把Connect、Operator这两个Profile拖到SOA_OPERATOR的Profiles下。



- 3、测试：用SOA_OPERATOR登录，Designer是进不去的，但Operator可以进去。

9.10.4. 例子：仅执行特定的方案

- 1、创建用户SOA_SCENARIO2，密码一样，把Connect这个Profile拖过来
- 2、

9.11. 安装Metadata Navigator

以Tomcat 5.5.26为例。

9.11.1. 发布oracledimn.war

- 1、将ODI安装盘\setup\manual下的oracledimn.war拷贝到Tomcat的webapps目录下
- 2、如果Tomcat尚未启动，则启动它，将自动解压并发布应用oracledimn

9.11.2. 额外配置

- 1、安装Oracle JDBC：下载并将Oracle JDBC的jar文件放到Tomcat的webapps\oracledimn\WEB-INF\lib下

注：可直接从本文档的附件中获得oracle_jdbc14.jar

- 2、配置连接信息：将ODI安装目录\oracledi\bin下的snps_login_work.xml拷贝到Tomcat的webapps\oracledimn\WEB-INF下

- 3、重新启动Tomcat

9.11.3. 登录并使用Metadata Navigator

- 1、登录地址：http://<host_name>:<port>/oracledimn



例子：<http://huajhua:8080/oracledimn>，用户SUPERVISOR，密码SUNOPSIS。

- 2、主要的功能都有，不过除了方案的执行和失败会话的重新启动，其他都只能浏览

注：失败会话还可以通过Operator、Web Service、OS命令重新启动。

Oracle Metadata Navigator										
会话列表										
状态	会话 ID	会话名称	用户	开始日期	结束日期	返回代码	持续时间(秒)	正在运行的子会话	成功子会话数	错误子会话
✓	135001	POP_EMP_MSSQL2ORACLE	SUPERVISOR	10/04/2009 17:12:24	10/04/2009 17:12:27	0	2	0	0	0
✗	134001	POP_EMP_MSSQL2ORACLE	SUPERVISOR	10/04/2009 17:11:03	10/04/2009 17:11:17	955	14	0	0	0
✓	133001	POP_EMP2MSSQL	SUPERVISOR	10/04/2009 16:59:58	10/04/2009 17:00:02	0	4	0	0	0
✓	128001	DEPT	SUPERVISOR	11/03/2009 16:14:38	11/03/2009 16:14:40	0	1	0	0	0
✓	127001	DEPT	SUPERVISOR	11/03/2009 16:14:31	11/03/2009 16:14:33	0	1	0	0	0
✓	126001	DEPT	SUPERVISOR	11/03/2009 16:14:24	11/03/2009 16:14:25	0	1	0	0	0
✓	125001	DEPT	SUPERVISOR	11/03/2009 16:14:16	11/03/2009 16:14:18	0	1	0	0	0
✓	124001	DEPT	SUPERVISOR	11/03/2009 16:14:09	11/03/2009 16:14:10	0	1	0	0	0
✓	123001	DEPT	SUPERVISOR	11/03/2009 16:14:01	11/03/2009 16:14:03	0	1	0	0	0
✓	122001	DEPT	SUPERVISOR	11/03/2009 16:13:54	11/03/2009 16:13:55	0	1	0	0	0
✓	121001	DEPT	SUPERVISOR	11/03/2009 16:13:47	11/03/2009 16:13:49	0	1	0	0	0
✓	120001	DEPT	SUPERVISOR	11/03/2009 16:13:40	11/03/2009 16:13:41	0	1	0	0	0
✓	119001	DEPT	SUPERVISOR	11/03/2009 16:13:33	11/03/2009 16:13:34	0	1	0	0	0
✓	118001	DEPT	SUPERVISOR	11/03/2009 16:13:25	11/03/2009 16:13:27	0	1	0	0	0
✓	117001	DEPT	SUPERVISOR	11/03/2009 16:13:18	11/03/2009 16:13:19	0	1	0	0	0
✓	116001	DEPT	SUPERVISOR	11/03/2009 16:13:11	11/03/2009 16:13:12	0	1	0	0	0
✓	115001	DEPT	SUPERVISOR	11/03/2009 16:13:03	11/03/2009 16:13:05	0	1	0	0	0
✓	114001	DEPT	SUPERVISOR	11/03/2009 16:12:56	11/03/2009 16:12:57	0	1	0	0	0
✓	113001	DEPT	SUPERVISOR	11/03/2009 16:12:48	11/03/2009 16:12:50	0	1	0	0	0
✓	112001	DEPT	SUPERVISOR	11/03/2009 16:12:41	11/03/2009 16:12:43	0	1	0	0	0
✓	111001	DEPT	SUPERVISOR	11/03/2009 16:12:32	11/03/2009 16:12:36	0	3	0	0	0
✓	110001	DEPT	SUPERVISOR	11/03/2009 16:12:22	11/03/2009 16:12:25	0	2	0	0	0
✓	109001	DEPT	SUPERVISOR	11/03/2009 16:12:10	11/03/2009 16:12:16	0	6	0	0	0
✓	108001	DEPT	SUPERVISOR	11/03/2009 14:20:13	11/03/2009 14:20:19	0	6	0	0	0

9.12. 安装Lightweight Designer

以Tomcat 5.5.26为例。

9.12.1. 发布oracledilwd.war

- 1、将ODI安装盘\setup\manual下的oracledilwd.war拷贝到Tomcat的webapps目录下
- 2、如果Tomcat尚未启动，则启动它，将自动解压并发布应用oracledilwd

9.12.2. 额外配置

- 1、安装Oracle JDBC：下载并将Oracle JDBC的jar文件放到Tomcat的/common/lib下

注：可直接从本文档的附件中获得oracle_jdbc14.jar

- 2、在Tomcat安装目录/conf/Catalina/localhost/oracledilwd.xml添加数据源和连接信息：

```
<Resource name="jdbc/ORACLE_MASTER" type="javax.sql.DataSource"
auth="Container"
driverClassName="oracle.jdbc.OracleDriver"
url="jdbc:oracle:thin:@HUAJHUA:1522:ORCL"
username="snpm"
password="snpm"
maxActive="8"
maxIdle="2"
maxWait="15000"
poolPreparedStatements="true"
removeAbandoned="true"
initialSize="2"
removeAbandonedTimeout="1800"/>
```

```
<Resource name="jdbc/ORACLE_WORK" type="javax.sql.DataSource"
  auth="Container"
  driverClassName="oracle.jdbc.OracleDriver"
  url="jdbc:oracle:thin:@HUAJHUA:1522:ORCL"
  username="snpw"
  password="snpw"
  maxActive="8"
  maxIdle="2"
  maxWait="15000"
  poolPreparedStatements="true"
  removeAbandoned="true"
  initialSize="2"
  removeAbandonedTimeout="1800"/>
```

- 3、在Tomcat安装目录webapps/oracledilwd/WEB-INF/web.xml添加资料库连接信息：

```
<resource-ref>
  <description>Oracle Datasource for the Master Repository</description>
  <res-ref-name>jdbc/ORACLE_MASTER</res-ref-name>
  <res-type>javax.sql.DataSource</res-type>
  <res-auth>Container</res-auth>
  <res-sharing-scope>Unshareable</res-sharing-scope>
</resource-ref>
<resource-ref>
  <description>Oracle Datasource for the Work Repository</description>
  <res-ref-name>jdbc/ORACLE_WORK</res-ref-name>
  <res-type>javax.sql.DataSource</res-type>
  <res-auth>Container</res-auth>
  <res-sharing-scope>Unshareable</res-sharing-scope>
</resource-ref>

<listener>
  <listener-
class>org.apache.myfaces.webapp.StartupServletContextListener</listener-class>
</listener>
```

- 4、在Tomcat安装目录webapps/oracledilwd/repositories.xml添加工作资料库配置：

```
<login name="Repository">
  <master name="jdbc/ORACLE_MASTER"
masterDriver="oracle.jdbc.OracleDriver"/>
  <work name="jdbc/ORACLE_WORK" workName="WORKREP1"/>
</login>
```

- 5、重新启动Tomcat

9.12.3. 登录并使用Lightweight Designer

- 1、登录地址：http://<host_name>:<port>/oracledilwd

例子：<http://huajhua:8080/oracledilwd>，用户SUPERVISOR，密码SUNOPSIS。

- 2、仅可以编辑Interface和Scenarios

10. 专题

10.1. DBLink

10.2. 对象冲突

11. Open and Closed Issues for this Deliverable

Add open issues that you identify while writing or reviewing this document to the open issues section. As you resolve issues, move them to the closed issues section and keep the issue ID the same. Include an explanation of the resolution.

When this deliverable is complete, any open issues should be transferred to the project- or process-level Risk and Issue Log (PJM.CR.040) and managed using a project level Risk and Issue Form (PJM.CR.040). In addition, the open items should remain in the open issues section of this deliverable, but flagged in the resolution column as being transferred.

Open Issues

ID	Issue	Resolution	Responsibility	Target Date	Impact Date

Closed Issues

ID	Issue	Resolution	Responsibility	Target Date	Impact Date