

ORACLE数据仓库相关组件 (OWB、OLAP、ODM)

使 用 手 册

作 者：王洋 (Ocean)
创建日期：2007-07-27

神州数码（中国）有限公司

目录

前言	V
第一部分 ORACLE WAREHOUSE BUILDER	1
第1章 WAREHOUSE BUILDER简介	1
第2章 WAREHOUSE BUILDER安装	3
2.1. OWB架构和组件	3
2.1.1. 基本概念	3
2.2. 安装需求	5
2.2.1. 服务器的需求	5
2.2.2. 数据库的需求	6
2.2.3. 客户机的需求	8
2.3. 安装步骤	8
2.4. 集群环境下的安装	11
第3章 安装配置OWB REPOSITORY	13
3.1. 启动REPOSITORY ASSISTANT	13
3.2. 安装REPOSITORY选项	13
3.2.1. 基本安装	14
3.2.2. 高级设置	14
3.3. 配置REPOSITORY步骤	19
第4章 卸载WAREHOUSE BUILDER	23
4.1 删除REPOSITORY用户	23
4.2 删除REPOSITORY所有者	24
4.3 卸载WAREHOUSE BUILDER软件	24
4.4 删除模式对象	25
第5章 安装的诊断和调试	26
5.1 日志文件的位置	26
5.2 不能启动任何WAREHOUSE BUILDER客户端或者其他产品	26
5.3 WAREHOUSE BUILDER挂起	26
5.4 额外一些有用的诊断日志	26
第6章 安全性简述	28
第7章 ORACLE WAREHOUSE BUILDER的使用	30
7.1. 设置 WAREHOUSE BUILDER	30
7.1.1. Projects	30
7.1.2. Modules	30
7.1.3. Collections	30
7.1.4. 选项设置	30
7.1.5. Validation	31
7.2. 设计 SOURCE MODULE	31
7.3. 设计 TARGET MODULE (TARGET SCHEMAS)	33
7.3.1. 概念介绍	33
7.3.2. 维 (Dimension)	35

7.3.3.	立方 (Cube)	39
7.3.4.	创建映射 (mappings)	40
7.3.5.	定义过程流 (Process Flows) 可选	41
7.3.6.	转换 (Transformation)	42
7.4.	发布到TARGET SCHEMAS (DEPLOY)	43
第二部分 ONLINE ANALYTICAL PROCESSING		45
第1章	OLAP概述	45
1.1.	拓扑	45
1.2.	解决的问题	45
1.3.	访问工具	45
1.4.	OLAP组件	46
1.5.	逻辑多维模型	47
1.6.	参数配置	47
1.7.	OLAP用户	48
第2章	操作ANALYTIC WORKSPACE	49
2.1.	ANALYTIC WORKSPACE MANAGER	49
2.2.	ANALYTIC WORKSPACE MANAGER的启动	50
2.3.	配置环境	50
2.4.	配置用户	50
2.5.	创建ANALYTIC WORKSPACE	50
2.6.	创建逻辑DIMENSIONS	52
2.6.1.	创建Dimension名称	52
2.6.2.	创建levels	53
2.6.3.	创建Hierarchies	54
2.6.4.	创建Attributes	55
2.7.	创建逻辑CUBES	55
2.7.1.	创建Cube名称	55
2.7.2.	创建Measures	56
2.7.3.	创建Calculated Measures(可选)	57
2.8.	映射逻辑对象到源数据	57
2.8.1.	映射dimension	58
2.8.2.	映射Cube	58
2.9.	加载聚合DIMENSION和CUBE	58
2.10.	查看DIMENSION和CUBE数据	60
2.10.1.	使用Analytic Workspace Manager	60
2.10.2.	使用OracleBI Spreadsheet Add-In	63
第三部分 DATA MINING		68
第1章	ODM概述	68
1.1.	DATA MINING 函数	68
1.2.	DATA MINING的数据准备	68
1.3.	SUPERVISED DATA MINING (PREDICTIVE)	69
1.4.	UNSUPERVISED DATA MINING (PREDICTIVE)	69
1.5.	DATA MINING过程	69
第2章	ODM的安装	70
2.1.	添加DATA MINING OPTION到数据库	70
2.2.	卸载DATA MINING OPTION	74

2.3.	升级DATA MINING OPTION	75
2.3.1.	删除旧版本的DM对象.....	75
2.3.2.	重建必要的DM视图.....	75
第3章	管理DM	75
3.1.	MICROSOFT WINDOWS的本地管理	75
3.2.	LINUX的本地管理	76
3.3.	远程管理.....	76
3.4.	创建DATA MINING用户	76
3.5.	导出导入DATA MINING模型	77
第4章	DM工具安装及使用.....	78
4.1.	ORACLE DATA MINER	78
4.2.	ORACLE SPREADSHEET ADD-IN FOR PREDICTIVE ANALYTICS	78
附录A	OWB各种工具启动方法	80
附录B	OLAP活动目录视图	81
附录C	OLAP动态性能视图	83

前言

本笔记总共包含以下三部分内容

- 第一部分 **Oracle Warehouse Builder**
- 第二部分 **OnLine Analytical Processing**
- 第三部分 **Oracle Data Mining**

通过这三部分的学习之后，基本对数据仓库会有一个很清晰的认识，另外本学习笔记主要是从纯技术角度来描述几个关键性组件的使用。学习过了之后，基本的配置使用，是肯定没有问题的，至于再深入的知识，需要进一步学习了。

参考文档：

《Oracle® Warehouse Builder User's Guide 10g Release 2 (10.2.0.2) 》

《Oracle® Warehouse Builder Installation and Administration Guide 10g Release 2 (10.2.0.2) for Windows and UNIX》

《Oracle® Warehouse Builder Transformation Guide 10g Release 2 (10.2)》

《Oracle® OLAP Reference 10g Release 2 (10.2)》

《Oracle® OLAP Application Developer's Guide 10g Release 2 (10.2)》

《Oracle® OLAP DML Reference 10g Release 2 (10.2)》

《Oracle® Data Mining Concepts 10g Release 2 (10.2)》

《Oracle® Data Mining Administrator's Guide 10g Release 2 (10.2)》

《Oracle® Data Mining Application Developer's Guide 10g Release 2 (10.2)》

第一部分 Oracle Warehouse Builder

第1章 Warehouse Builder简介

Oracle Warehouse Builder (OWB) 提供了端对端数据管理的企业级的解决方案，是一个简单而又覆盖广泛的数据管理工具。

Oracle Warehouse Builder帮助用户设计、部署与管理数据仓库。是 Oracle 用于设计与部署数据仓库解决方案的技术，为设计、部署企业数据仓库数据集市和电子商务智能应用程序的可扩展框架提供集成

使用Oracle Warehouse Builder图形化界面，可以快速设计、部署数据仓库。用户在建立数据仓库的设计过程中，可以使用向导驱动程序指导我们完成具体的设计过程。

元数据源定义的向导驱动过程支持从已有元数据源向Oracle Warehouse Builder知识库的导入。Oracle Warehouse Builder支持3NF和星型模式的设计，可以从Oracle Designer中导入现存设计。同时，Oracle Warehouse Builder还为事实表和维度提供向导及图形编辑器。

自动生成代码，OWB可以自动生成编码，并且通过校验程序保证编码的正确性，按照部署的要求生成不同的编码类型。

用户通过Oracle Library可以访问所有内置在Oracle Warehouse Builder中的转换。OWB导出所有可应用的数据库函数作为转换。它也包括预先定义的函数和过程，用于转换或者在映射触发器中执行普通仓库函数。它的结构适合于按照转换类型快速访问的情况，每个转换的在线说明可以指导用户进行正确的转换。

Oracle Warehouse Builder全面应用Oracle的分区、索引和总结管理等特性。与数据库的紧密集成允许Oracle作为一种转换引擎使用，排除了增加转换服务器的需求。

Oracle Warehouse Builder 主要特征在于数据的固化和整合：

Oracle Warehouse Builder提供接入各种数据源的接口，包括：各种第三方数据库，应用，平面文件，excel表格等等。

第2章 Warehouse Builder 安装

2.1. OWB架构和组件

2.1.1. 基本概念

◆ Design Center

Design Center 是一个图形化的工具，良好的图形用户界面，主要用来进行源的定义，目标方案的设计以及ETL过程的处理。

我们用Design Center来设计、管理、制定和部署ETL过程。

所有设计过程中的元数据都存储在Warehouse Builder repository中。

◆ Control Center Manager

我们需要在Control Center Manager管理里进行部署和执行特定的ETL过程，是一个全面的部署发布控制台，我们可以通过Control Center Manager了解到ETL部署发布的过程细节。

◆ Target Schema

从字面意思上就可以理解，目标方案或者叫做目标模式，是构建数据仓库时，我们要加载数据的目标模式，我们目标就是要把在Design Center中设计的数据对象（比如立方，维度，视图和映射等等）全部加载到目标模式中。

Target Schema 不是一个Warehouse Builder的组件，它是数据库中的一个组件，简单的说，就是数据库中的一个模式（schema）。

◆ Warehouse Builder Repository

一个Warehouse Builder Repository是由一个repository所有者、一个或者多个 repository用户、一个单独的（可选）Control Center模式组成。Repository 所有者储存所有源、目标以及ETL过程的定义的元数据。除了储存设计时的元数据，还包含由Control Center Manager和Control Center Service产生的运行时元数据。

可以使用Repository Assistant图形工具来定义和管理一个或者多个repositories。

一个或者多个Target Schema对应于一个Warehouse Builder Repository。

◆ Warehouse Builder Repository Owner

◆ Warehouse Builder Repository User

Warehouse Builder repository由几个部分组成：**repository owner**，一个或者多个**repository user**。repository schema 用来存储所有的源，目标以及ETL过程的定义元数据。一个Repository除了包含设计过程中的元数据卡，还包含由Control Center Manager 和 Control Center Service产生的运行元数据。

repository owner拥有所有的管理权限，包括管理repository和显示语言和用户。

repository user可以创建一个或者多个，共享一个Repository的元数据来进行各自的功能实现。

◆ Repository Browser

Repository Browser是一个WEB的接口，通过Repository Browser，我们可以查看查看repository 的元数据，并生成相应的报告，需要配合着application server来使用。

◆ Control Center Service

Control Center Service是Warehouse Builder的一个组件，可以理解为一个服务，有了这个服务，我们才可以注册locations，才可以通过Control Center Manager来发布部署和执行ETL过程等。

◆ Mapping

用来定义从源数据到目标数据的一个过程，通过这个过程设计，OWB生成相应的过程代码。

◆ Deployment

是一个部署源代码的过程，这个过程中，OWB复制有关的元数据和生成的mapping代码到目标模式（**Target Schema**）里，在Target Schema里会执行在**Desing Center**里设计出来的ETL逻辑。

◆ **Locations**

存储连接信息，这个连接包括平面文件，数据库或者OWB来提取或者加载数据的第三方应用等等，这些连接可以是数据源的连接，也可以是数据目标的连接。

◆ **Modules**

是包含在一个project中的一组对象的集合

2.2. 安装需求

2.2.1. 服务器的需求

◆ **UNIX 服务器**

所有的UNIX服务器（除了linux）上只能安装Warehouse Builder服务端组件，也就是说，我们可以在UNIX服务器上安装Repository和 Control Center Service，但是Design Center 和Repository Browser就另外需要 windows或者linux平台的服务器了。

在linux和windows平台上，既可以安装服务端组件又可以安装客户端组件。

需求	最低值
磁盘空间	1100 MB
内存	768 MB
交换空间	1 GB

设置环境变量

Environmental Variable	ORACLE_HOME
C Shell	setenv ORACLE_HOME full_path
Korn Shell	export ORACLE_HOME=full_path
Bourne Shell	ORACLE_HOME=full_path; export ORACLE_HOME

◆ WINDOWS 服务器

WINDOWS 平台服务器可以安装Warehouse Builder服务端组件，也可以安装Warehouse Builder客户端组件

需求	最低值
磁盘空间	1100 MB
内存	768 MB
交换空间	1 GB

2.2.2. 数据库的需求

◆ 版本要求

Warehouse Builder 10g Release 2 (10.2.0.2)支持下面版本的数据库：

- Oracle9i Release 2 (9.2.x) Enterprise Edition
- Oracle Database 10g Enterprise Edition R1 (10.1.x)
- Oracle Database 10g Standard Edition R2 (10.2.x)
- Oracle Database 10g Enterprise Edition R2 (10.2.x)

◆ 初始化参数配置的要求

1. Design Repository Database Instance

初始化参数	值	备注
COMPATIBLE	10.2.0.1	
DB_BLOCK_SIZE	8192	Warehouse Builder 不建议超过 8K 的值 design repository.
LOCK_SGA	TRUE	
OPEN_CURSORS	300	大于 300.

2. Runtime Repository Database Instance

初始化参数	值	备注
AQ_TM_PROCESSES	1	大于 1
DB_BLOCK_SIZE	16384	建议值是 16384. 使用尽可能系 If your computer has less than 512 MB of RAM, a value of 9600 is recommended.
DB_FILE_MULTIPLE_BLOCK_READ_COUNT	16	16 建议值, 32 最佳值
ENQUEUE_RESOURCES	3000	至少 3000, 如果需要导入大 MDL 文件的话, 需要设置的更高
JOB_QUEUE_PROCESSES	大于 10	
MAX_COMMIT_PROPAGATION_DELAY	0	RAC 环境必须的. If it is not set to 0, then data propagation delays may cause NO_DATA_FOUND errors in the Control Center Service.
OPTIMIZER_MODE	all_rows	For other possible optimizer modes, see <i>Oracle Designing and Tuning for Performance</i> , <i>Oracle Database Performance Tuning Guide and Reference</i> , and <i>Oracle Data Warehousing Guide</i> .
PARALLEL_ADAPTIVE_MULTI_USER	TRUE	Set PARALLEL_AUTOMATIC_TUNING to TRUE as a prerequisite for this parameter.
QUERY_REWRITE_ENABLED	TRUE	若使用 QUERY REWRITE 选项的物化视图, 必须要设置这个

◆ 为平面文件目标设置路径

为了配置平面文件目标, 需要配置目标文件路径, 也就是设置 [UTL_FILE_DIR](#) 参数

这个参数支持多重配置，比如设定两个合法的路径的语句如下：

```
UTL_FILE_DIR = D:\Data\FlatFiles
UTL_FILE_DIR = E:\OtherData
```

两行之间必须是连续的，中间不能再穿插其他初始化参数

另外也可以设置为'*'，允许所有目录有效

2.2.3. 客户机的需求

◆ LINUX 服务器

需求	最低值
磁盘空间	1100 MB
内存	768 MB
交换空间	1 GB

设置环境变量

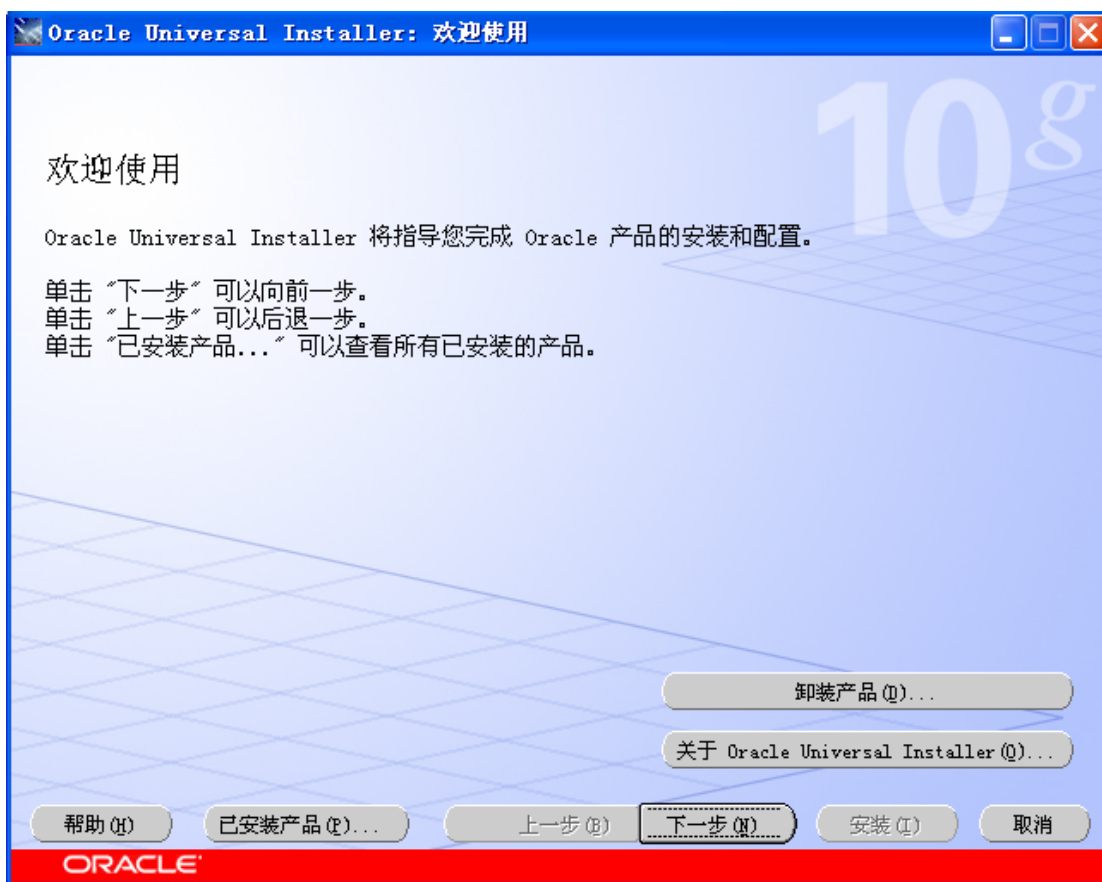
Environmental Variable	ORACLE_HOME
C Shell	setenv ORACLE_HOME full_path
Korn Shell	export ORACLE_HOME=full_path
Bourne Shell	ORACLE_HOME=full_path; export ORACLE_HOME

◆ WINDOWS 服务器

需求	最低值
磁盘空间	850 MB
内存	768 MB
交换空间	1 GB

2.3. 安装步骤

1. 决定实施策略，几种可行的实施策略的方案，参见<安装Repository选项>小节。
2. 准备服务器的需求，参见<安装需求－服务器的需求>小节。
3. 如果是集群环境，参见<集群环境下的安装>小节，否则下一步。
4. 数据库的需求，参见<安装需求－数据库的需求>小节。
5. 开始安装 Warehouse Builder软件。







6. 创建Warehouse Builder Repository。详见<第3章>
7. 设置安全策略(可选)，详见第3章
8. 安装可选组件，详见第3章
9. 安装客户端Warehouse Builder软件。
10. 安装完毕，可以启动相关组件进行相应的功能实施了。

2.4. 集群环境下的安装

1. 首先按照基本的安装步骤，正确的安装完毕 ClusterWare 和 database 软件。
2. 主机配置
为每一个节点配置目录 OWB_ORACLE_HOME\network\admin 下面的 tnsnames.ora 文件，
3. 关键参数 MAX_COMMIT_PROPAGATION_DELAY，要设置为 0。

4. 参考单机的安装步骤进行安装 Warehouse Builder 软件。
5. 安装 Warehouse Builder Repository
6. Register each RAC node.
每一个节点都分别运行 Repository Assistant, 选择高级配置, 进行实例的注册。
7. 复制\$OWB_ORACLE_HOME/owb/bin/admin/rtrepos.properties 文件到 RAC 集群的每一个节点
8. 为 Repository 设置安全策略 (可选).
9. 安装选件 (可选).
10. 安装客户机软件
11. 安装结束

注：在 RAC 环境下，建议在每个节点上分别进行单机模式的安装。

第3章 安装配置OWB Repository

通常我们使用Repository Assistant 去安装定义和管理 Warehouse Builder repository, 还有一种OMB Plus脚本语言也可以用来定义和管理 Warehouse Builder repository, 实质上Repository Assistant图形工具底层也是调用一系列的OMB Plus脚本接口语法, 这个脚本语言在本书中不作详细介绍, 详细的有关OMB Plus脚本语言的使用语法, 请参考详细的《Oracle Warehouse Builder Scripting Reference》, 本书主要介绍Repository Assistant工具的使用。

3.1. 启动Repository Assistant

Windows平台:

开始, 程序, OWB_ORACLE_HOME, administration, Repository Assistant

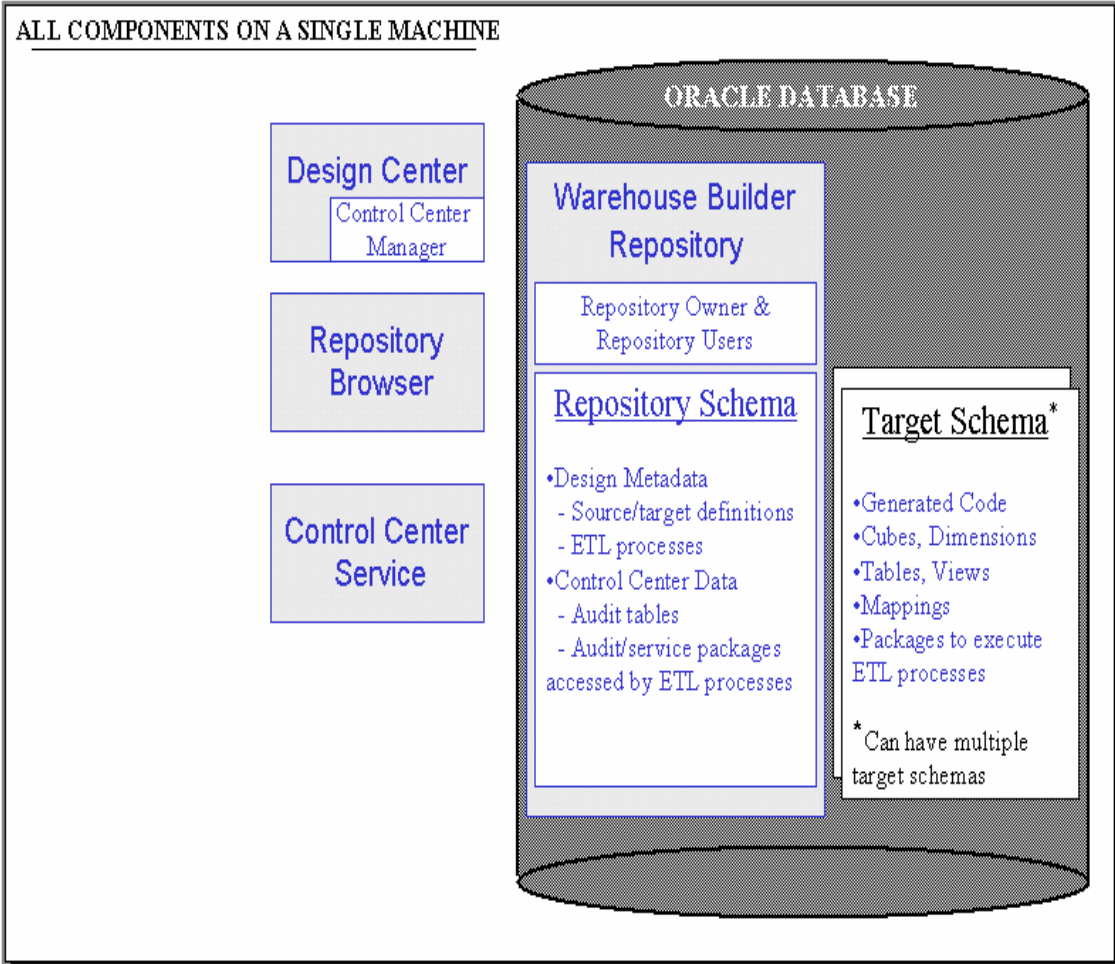
Linux平台:

```
$OWB_ORACLE_HOME/owb/bin/unix/reposinst.sh
```

3.2. 安装Repository选项

3.2.1. 基本安装

Figure 2-1 Consultant Profile: Basic Installation on a Single Computer



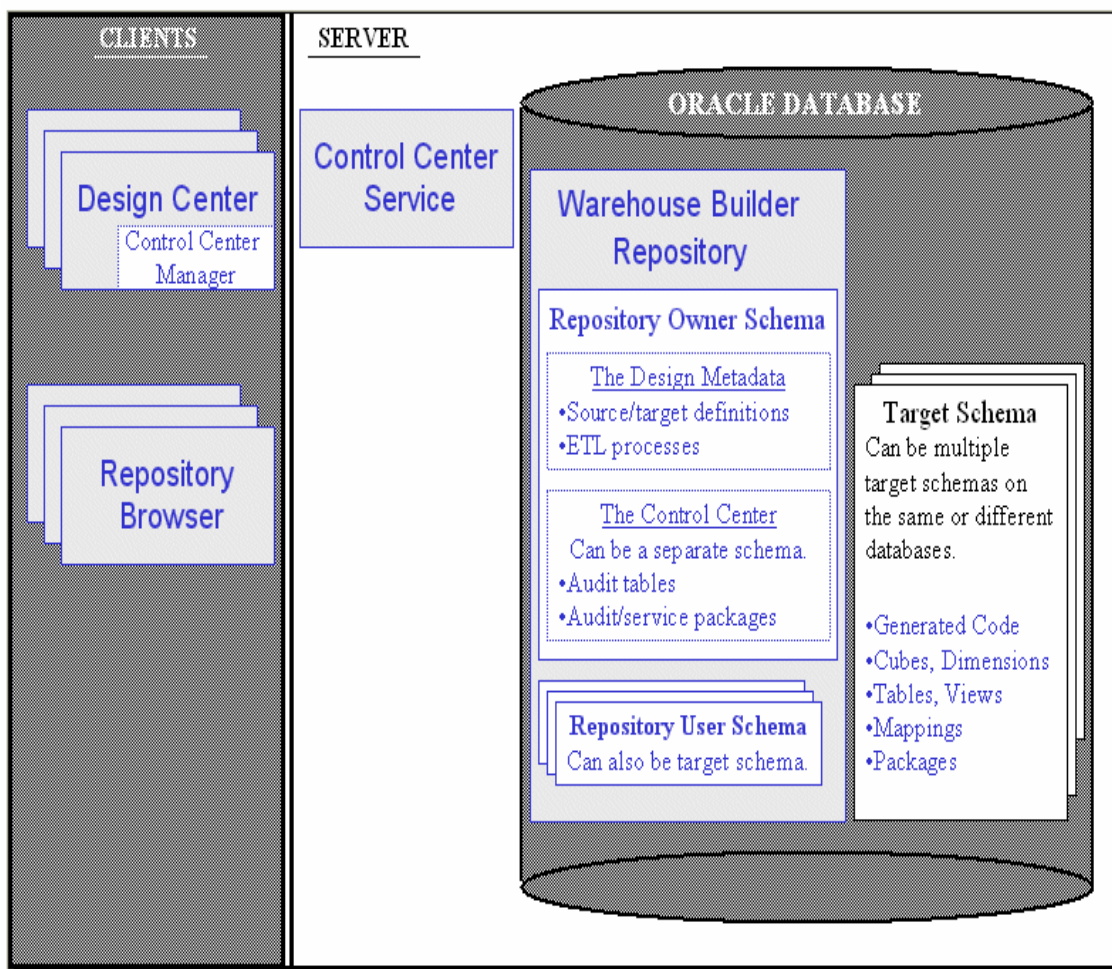
3.2.2. 高级设置

高级设置选项里，又分为三种高级设置方式：

3.2.2.1 传统的client/server模式

多个clients访问同样的一个repository，也可以为每一个client定义一个repository，基本结构图如下：

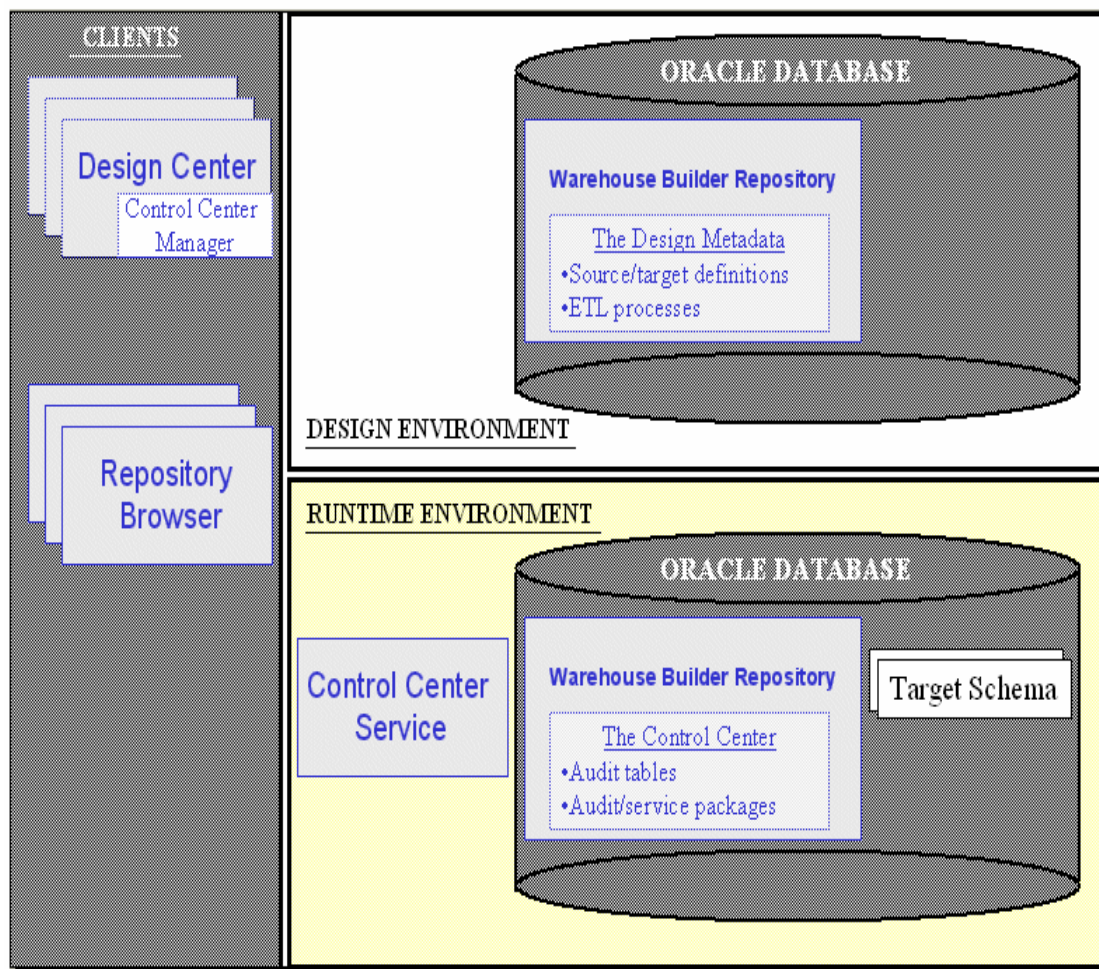
Figure 1-1 Warehouse Builder with Traditional Client/Server Implementation



3.2.2.2 分离的design和runtime模式

顾名思义，design阶段的repository和runtime节点的repository分别安装在不同的数据库里，基本结构图如下：

Figure 1-2 Split Repositories Implementation



3.2.2.3 远程runtime模式

在一个不安装任何Warehouse Builder组件的一个目标环境里部署一个target schema。这种环境下的配置，必须确保Control Center Service在另一个主机上运行着。

这种模式下，就能实现将target schema创建在Warehouse Builder不支持的平台上，比如：Warehouse Builder 10g Release 2 (10.2.0.2)不支持将Control Center运行在HP OpenVMS Alpha平台上，这种情况下，就可以利用远程runtime模式来将Control Center运行在另外一台主机上。

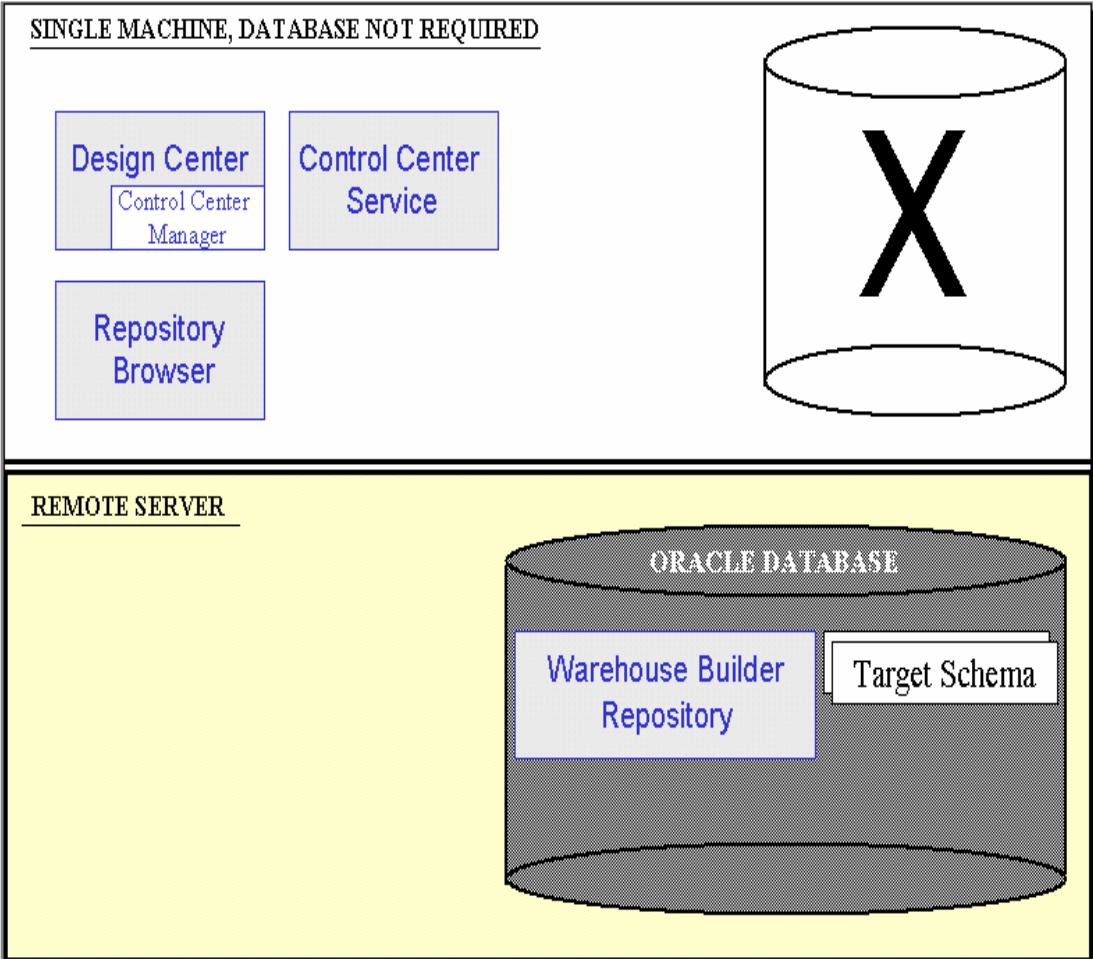
远程runtime模式有三种配置方法：

◆Control Center Service安装在Client机上：

运行Control Center Service 的主机上不需要安装oracle 数据库，这种配置方式下，可以没有任何约束的发布所有类型的mapping到远程的target里。

基本结构图如下：

Figure 2-4 Control Center Service Installed on the Client

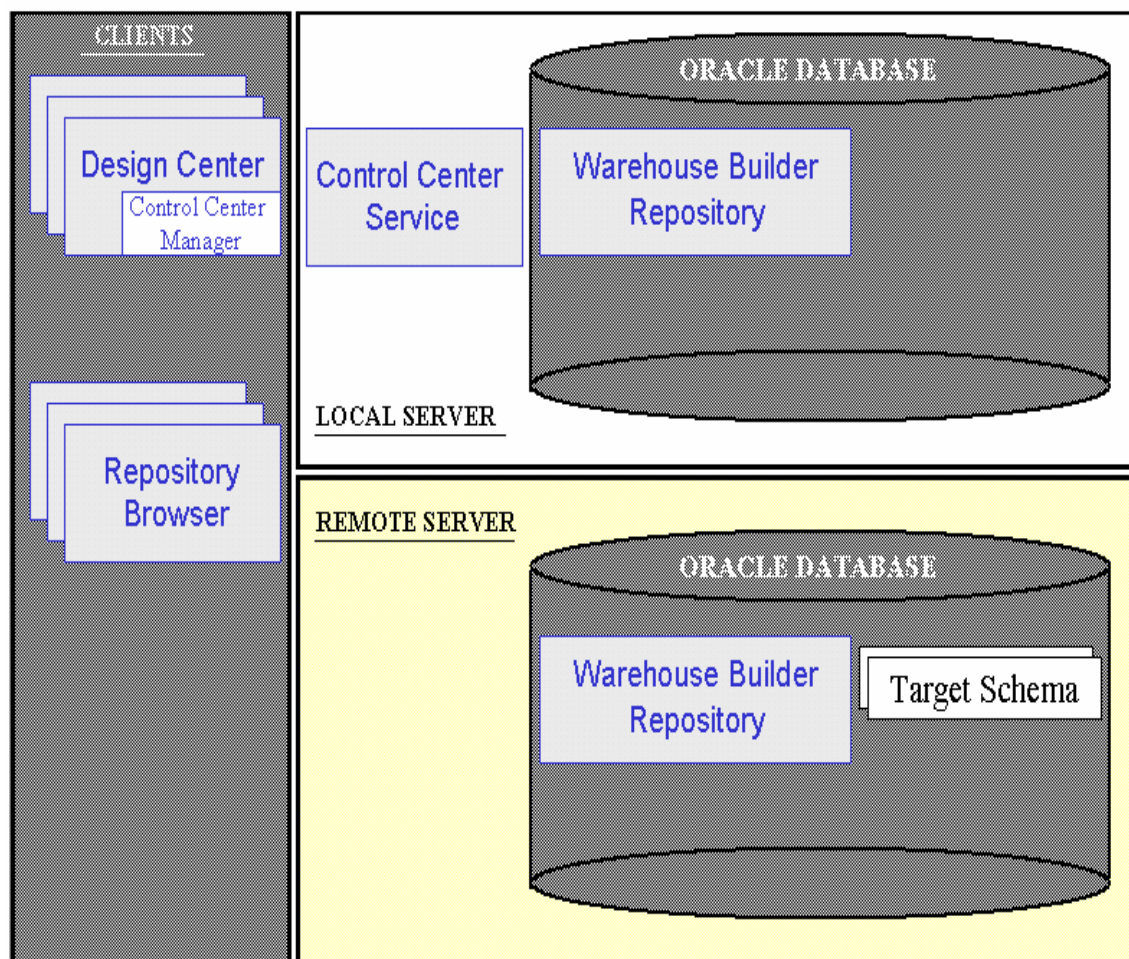


◆Control Center Service安装在一个本地的服务器上：

将Control Center Service安装在一个本地的服务器上，这种配置方式下，可以没有任何约束的发布所有类型的mapping到远程的target里。

基本结构图如下：

Figure 2-5 Control Center Service Installed on a Local Server

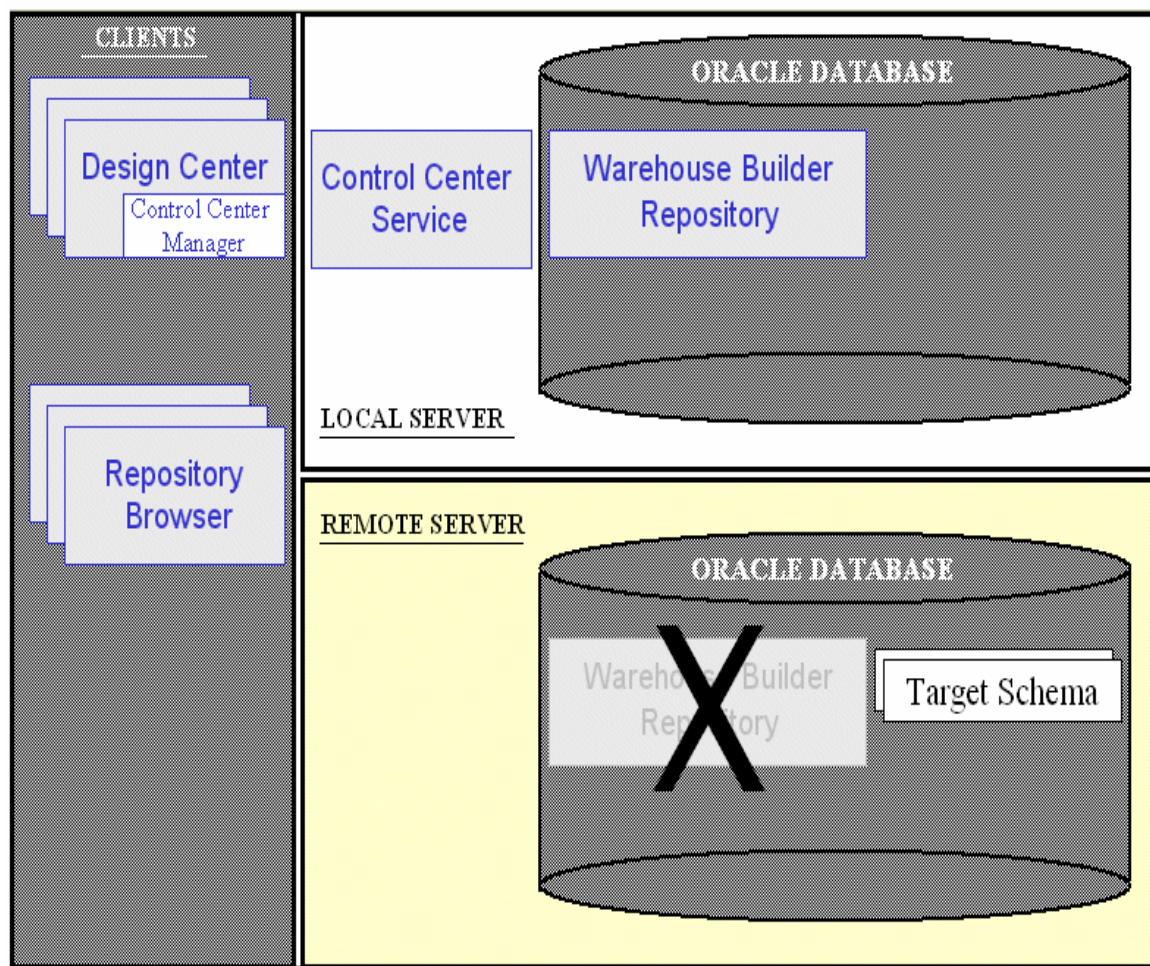


◆A Standalone Target Schema:

这种模式下, 在目标数据库中, 只有一个target schema, 没有Warehouse Builder Repository任何信息。这种配置模式, 在发布mapping时有一些限制, 不能发布PL/SQL类型的mapping到target schema里面。

基本结构图如下:

Figure 2-6 A Standalone Target Schema



3.3. 配置repository步骤

基本安装下比较简单，默认会创建一个repository owner (OWBRT_SYS)，也可以创建一个新的repository owner，这里主要介绍一下使用高级设置的配置步骤。

3.3.1 管理Warehouse Builder Repository Owner

◆ 创建新的repository owners

创建新的owner，需要选择默认表空间，设置密码等，按照向导进行创建。

选择基础语言, Warehouse builder会使用这个基础语言来对repository对象进行physical name的命名。基础语言只能在创建repository时指定, 创建完毕之后, 不能再进行基础语言的更改, 也不能再添加新的基础语言。

但是我们可以添加和改变显示语言, 显示语言主要是为了repository对象的business name, 可以添加很多的显示语言来支持每一个business name

目前Warehouse builder支持的显示语言如下:

ISOID	Language
sq_AL	Albanian
en_US	American English
ar_AE	Arabic
ar_EG	Arabic Egypt
as_IN	Assamese
bn_IN	Bangla
pt_BR	Brazilian Portuguese
bg_BG	Bulgarian
fr_CA	Canadian French
ca_ES	Catalan
hr_HR	Croatian
cs_CZ	Czech
da_DK	Danish
nl_NL	Dutch
en_GB	English
et_EE	Estonian
fi_FI	Finnish
fr_FR	French
de_DE	German
el_GR	Greek
gu_IN	Gujarati
he_IL	Hebrew
hi_IN	Hindi
hu_HU	Hungarian

ISOID	Language
is_IS	Icelandic
in_ID	Indonesian
it_IT	Italian
ja_JP	Japanese
kn_IN	Kannada
ko_KR	Korean
es_US	Latin American Spanish
lv_LV	Latvian
lt_LT	Lithuanian
ms_MY	Malay
ml_IN	Malayalam
mr_IN	Marathi
es_MX	Mexican Spanish
no_NO	Norwegian
or_IN	Oriya
pl_PL	Polish
pt_PT	Portuguese
pa_IN	Punjabi
ro_RO	Romanian
ru_RU	Russian
zh_CN	Simplified Chinese
sk_SK	Slovak
sl_SI	Slovenian
es_ES	Spanish
sv_SE	Swedish
ta_IN	Tamil
te_IN	Telugu
th_TH	Thai
zh_TW	Traditional Chinese
tr_TR	Turkish
uk_UA	Ukrainian

ISOID	Language
vi_VN	Vietnamese

◆ Dropping Repository Owners

要注意一点：一个repository owner会与很多repository user关联的，所以repository owner删除后，所有遗留的repository user就没有了关联，不能再使用Repository Assistant进行删除了，所以在删除repository owner之前一定要先删除相关联的repository user

3.3.2 管理Warehouse Builder Repository User

方法可以参考《管理Warehouse Builder Repository Owner》

第4章 卸载Warehouse Builder

4.1 删除Repository用户

删除 repository 所有者之前必须先删除 repository 用户，删除 repository 用户只是从 repository 里将用户删除，并不会对数据库用户进行任何的更改。

基本步骤：

1. 启动 Oracle Warehouse Builder Repository Assistant.

Windows平台：

开始，程序，OWB_ORACLE_HOME, administration, Repository Assistant

Linux平台：

```
$OWB_ORACLE_HOME/owb/bin/unix/reposinst.sh
```

2. 在安装类型页面上，选择高级设置。
3. 在连接描述页面，按提示输入各种信息：
 - SYSDBA User Name
 - SYSDBA Password
 - Host Name
 - Port Number
 - Oracle Service Name
4. 接下来，选择 Manage Warehouse Builder repository users
5. 再选择 Delete the registration of one or more Warehouse Builder repository users.
6. 选择所有者并提供相应信息.
7. 选择要删除的 repository 用户，点击右箭头，挪到选择用户里

8. 点击 **Finish**.

4.2 删除Repository所有者

删除 repository 用户之后就可以删除 repository 所有者了，删除 repository 所有者只是从 repository 里将用户删除，并不会对数据库用户进行任何的更改。

基本步骤：

1. 启动 Oracle Warehouse Builder Repository Assistant.

Windows平台：

开始，程序，OWB_ORACLE_HOME, administration, Repository Assistant

Linux平台：

`$OWB_ORACLE_HOME/owb/bin/unix/reposinst.sh`

执行《删除 Repository 用户》小节的前三步骤

2. 接下来，选择 Manage a Warehouse Builder repository owner
3. 再选择 Delete an existing Warehouse Builder repository owner
4. 选择所有者并提供相应信息.
5. 点击 Finish.

4.3 卸载Warehouse Builder软件

基本步骤：

1. 启动 Oracle Universal Installer.

Windows平台：

开始，程序，OWB_ORACLE_HOME, Oracle Installation Products, Universal Installer

Linux平台：

```
$OWB_ORACLE_HOME/oui/bin/runInstaller.sh
```

2. 在欢迎页面里，点击 Deinstall Products.
3. 选择要删除的 oracle_home.
4. 点击 Remove.
5. 点击 Yes
6. 点击 close
7. 点击 cancel 退出.

4.4 删除模式对象

执行完毕“删除 repository 用户”和“删除 repository 所有者”之后，只是从 repository 里将用户删除，并不会对数据库用户进行任何的更改，所以需要手工在数据库里的各种用户和相关的角色进行删除操作。

第5章 安装的诊断和调试

5.1 日志文件的位置

- **Warehouse Builder Repository Assistant**日志位置:

`$OWB_ORACLE_HOME\owb\UnifiedRepos\log_<timestamp>.log`:

- **Warehouse Builder Control Center Service**日志位置

`$OWB_ORACLE_HOME\owb\log\Repository_Name\log.xx`

- **Warehouse Builder Design Center**日志位置

在 Preferences 标签页里设置

5.2 不能启动任何Warehouse Builder客户端或者其他产品

发现启动 Warehouse Builder client 之后, 只是出现了一个初试界面就再也没有反应了, 等待很久也不会有错误提示出现, 就是一个启动界面挂起状态。

原因就是后续安装的 oracle 软件覆盖了必要的 java 对象。

建议最后安装 OWB 组件, 在安装完毕之后尽量不再安装 oracle 软件。

5.3 Warehouse Builder挂起

可以执行命令行语句查看错误信息

`owbclient.bat`.

5.4 额外一些有用的诊断日志

1. 转到

Windows: *OWB_ORACLE_HOME\owb\bin\win32*

UNIX: *OWB_ORACLE_HOME/owb/bin/unix*

2. 运行相应的命令，比如 `owbclient.bat > owbclient.log`
3. 查看生成的日志文件

第6章 安全性简述

略

第7章 Oracle Warehouse Builder的使用

7.1. 设置 Warehouse Builder

在设计目标数据系统之前，可能会做的一些设置工作，主要包含四个部分的说明，包括一些可选的操作。先了解一些必要的概念。

7.1.1. Projects

Projects是Warehouse Builder repository里面最大的一个存储对象，project组织相关的元数据定义，这些定义包括数据定义，mappings和transformation 操作。

7.1.2. Modules

Modules是在Warehouse Builder repository中，将一类设计对象集合起来的一个逻辑存储对象，在发布的时候，可以以Module为单位进行发布和日常的维护操作。

7.1.3. Collections

Collections是Warehouse Builder repository里面存储元数据的一个结构，在需要导出某些对象到其他工具或者系统里的时候，就需要创建一个Collection来进行这类对象元数据的存储。

7.1.4. 选项设置

- ◆ Appearance
- ◆ Control Center Monitor
- ◆ Data Profiling
- ◆ Deployment

- ◆ Environment
- ◆ Generation/Validation
- ◆ Logging
- ◆ Naming
- ◆ Security

7.1.5. Validation

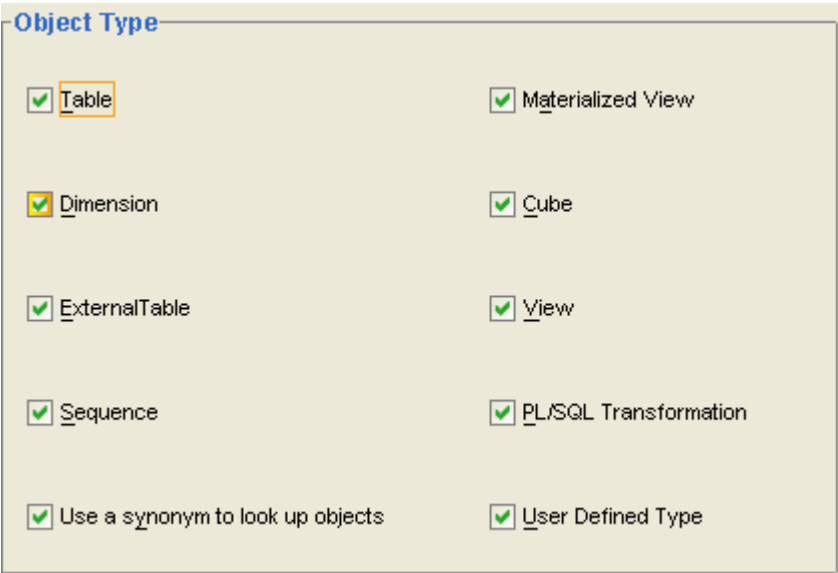
顾名思义，进行有效性验证的阶段，当我们对每个对象设计完毕之后，可以进行validation来对设计对象的有效性进行验证。

7.2. 设计 Source Module

定义源数据，导入元数据：

顾名思义：定义源数据是要通过工具来定位到我们预处理的源数据库或者源文件等，源数据是不能被更改的，它只是提供一个数据的来源。那么，我们要想对这个数据源进行相关的加工，就需要将其的元数据导入到数据仓库的模型当中来，在OWB中，这个模型就是Source Module。

目前OWB支持以下原始数据库中的数据类型：



Object Type	
<input checked="" type="checkbox"/> Table	<input checked="" type="checkbox"/> Materialized View
<input checked="" type="checkbox"/> Dimension	<input checked="" type="checkbox"/> Cube
<input checked="" type="checkbox"/> ExternalTable	<input checked="" type="checkbox"/> View
<input checked="" type="checkbox"/> Sequence	<input checked="" type="checkbox"/> PL/SQL Transformation
<input checked="" type="checkbox"/> Use a synonym to look up objects	<input checked="" type="checkbox"/> User Defined Type

另外还可以从平面文件中导入元数据，也可以从第三方的应用来导入数据。

目前Warehouse Builder 10.2 能够接入的存储系统 and 应用列表如下；

位置节点	支持的源数据	支持的目标
Databases/Oracle	Oracle db 8.1, 9.0, 9.2, 10.1, 10.2	Oracle db 9.2, 10.1, 10.2
Databases/Non-Oracle	Any database accessible through Oracle Heterogeneous Services, including but not limited to DB2, DRDA, Informix, SQL Server, Sybase, and Teradata. Any data store accessible through the ODBC Data Source Administrator, including but not limited to Excel and MS Access.	To load data into spreadsheets or third-party databases, first deploy to a comma-delimited or XML format flat file.
Files	Delimited and fixed-length flat files.	Comma-delimited and XML format flat files.
Applications	SAP R/3 3.x, 4.x Oracle E-Business Suite PeopleSoft 8, 9	none
Process Flows and Schedules/OEM	None	OEM Agent 9.0, 9.2
Process Flows and Schedules/Oracle Workflow	None	Oracle Workflow 2.6.2, 2.6.3, 2.6.4, 11i
Process Flows and Schedules/Concurrent Manager	None	Concurrent Manager 11i
Business Intelligence/BI Beans	None	BI Beans 10.1
Business Intelligence/Discoverer	None	Discoverer 10.1

7.3. 设计 Target Module(Target Schemas)

我们使用Data Object Editor来进行Target Schemas的设计工作，下面主要介绍一下在Target Schemas里面的各种数据库对象的设计，和Data Object Editor的使用。

由于数据库对象纷繁复杂，我们在这里选择几个比较典型的对象类型来进行相应功能的说明，其他未提及的对象方法比较类似，可以参考相关设计文档进行详细的设计。

7.3.1. 概念介绍

Warehouse Builder支持关系型 (relational) 和空间 (dimensional) 数据对象。

关系型对象依靠数据库中的表来存储数据，定义的关系对象就是在数据库中的不同对象来存储数据，关系型对象包括表，视图，物化视图，序列等等。

空间对象需要额外的一些元数据来定义数据存储，需要在一个结构格式下描述更多的逻辑关系来支持存储数据。空间对象包括维 (dimension) 和立方 (cube) 。

空间对象 (dimensional) 有三种实现方法

◆ Relational Implementation of Dimensional Objects

在数据库中，通常使用表来存储维对象定义及维的数据，对维的一个查询操作，实质上就是对表的数据提取。OWB创建其相应的DDL脚本语句，可以移植到其他的数据库当中。

在维的创建过程中，可以指定是否为预创建的维指定现有的数据库表来存储维的数据，如果不指定，可以在创建维的过程中，OWB自动为维创建必要的底层数据库表来存储数据。

这就涉及到几个概念：

1. 绑定 (Binding)

包括自动绑定和手工绑定两种方法。

自动绑定，是OWB在创建维的过程中默认采用的一种绑定方法，这样在创建完毕维之后，维就已经和底层的数据表建立了在维的属性和表的列之间的一个关系。

手工绑定，是我们已经在数据库中存在一系列的表，这时我们再创建维的时候，可以选择将维绑定在已经存在的表上。

2. 卸载绑定 (Unbinding)

顾名思义，就是解除维和表之间的关联关系。

◆ ROLAP Implementation of Dimensional Objects

这种方式与relational implementation一样，也是将维的信息实际存储在数据库表中的，但是与relational implementation不同的是，在发布时，不是通过生成DDL脚本在其他数据库中执行来完成的，而是会创建CWM2 metadata for the dimensional object in the OLAP catalog。

◆ MOLAP Implementation of Dimensional Objects

这种方式下，维对象的数据是存储在分析空间中（analytic workspace）的，分析空间是OLAP中的一个概念，在OLAP相关章节中再做介绍。

下面分别详细的说明一下Dimension和Cube。

7.3.2. 维 (Dimension)

7.3.2.1 列举一堆基本要素:

一个维 (dimension) 必须有一个surrogate identifier和多个 business identifier.

Surrogate identifier 只包含一个属性, 但是 business identifier 可以包含多个属性

每一个维至少包含一个属性, 即surrogate identifier

一个维如果使用的是relational or ROLAP implementation, 那么至少需要包含一个层次 (level)

一个维如果使用的是relational or ROLAP implementation, 那么必须 associated with a sequence that is used to load the dimension key attribute.

The dimension key attribute of a dimension that uses a relational or ROLAP implementation must bind to the primary key of a table.

A Type 2 SCD must have the effective date, expiration date, and at least one triggering attribute.

A Type 3 SCD must have the effective date and at least one triggering attribute.

7.3.2.2 维的属性 (Attributes)

即维的一个成员的描述, 一个属性适用于一个或者多个级别 (level) , 他们作为级别的属性来存储数据。在创建维的时候, 要保证创建所有的各个级别所要用的属性, 维的属性是各种BI工具能看到的唯一属性。

7.3.2.3 级别 (level)

级别(level)体现的是整个维聚合的程度, 每一个级别有级别的属性和属性的标识。属性标识分为surrogate identifier 和 business identifier

7.3.2.4 层次 (Hierarchy)

组织级别排序的一个逻辑结构, 比如:

Fiscal Hierarchy:

Fiscal Year > Fiscal Quarter > Fiscal Month > Fiscal Week > Day

Calendar Hierarchy:

Calendar Year > Calendar Quarter > Calendar Month > Day

举一个维的例子

Products Dimension:

Level	Attribute Name	Identifier
Total	ID	Surrogate
	Name	Business
	Description	Business
Groups	ID	Surrogate
	Name	Business
	Description	Business
Product	ID	Surrogate
	UPC	Business
	Name	Business
	Description	
	Package Type	
	Package Size	

包含一个层次:

Hierarchy 1: Total > Groups > Product

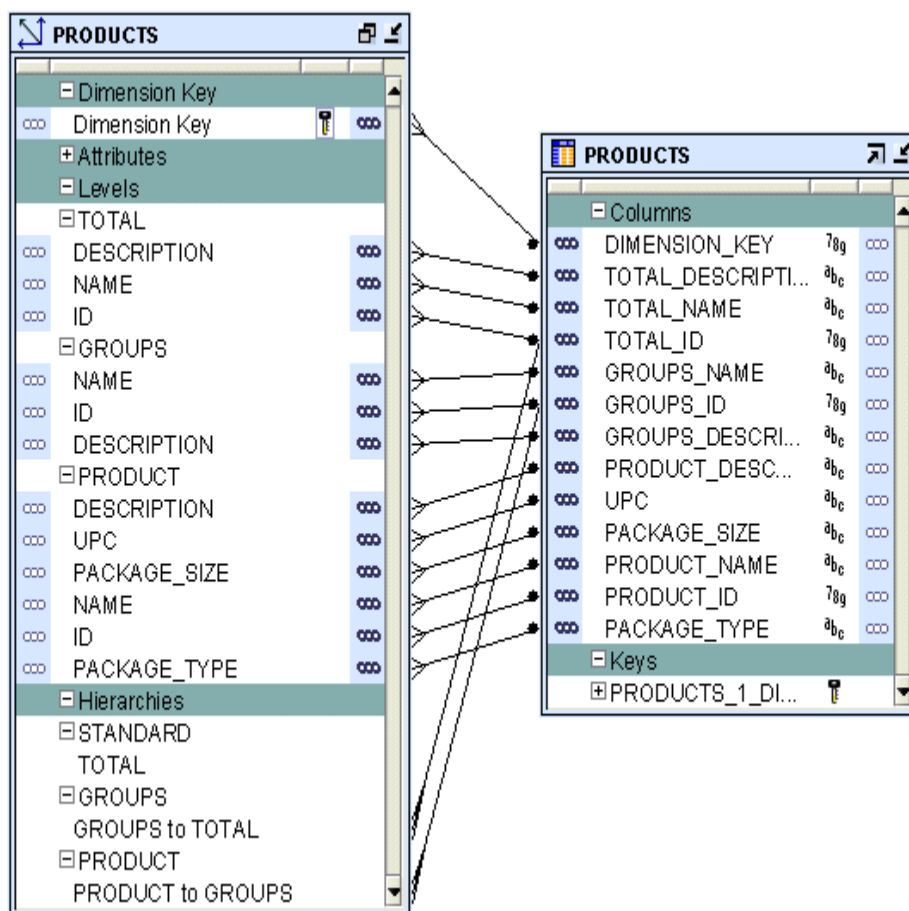
7.3.2.5 维的实现

维的实现，也就是维的数据存储问题，有两种方式，星型（star）模式和雪花（snowflake）模式。

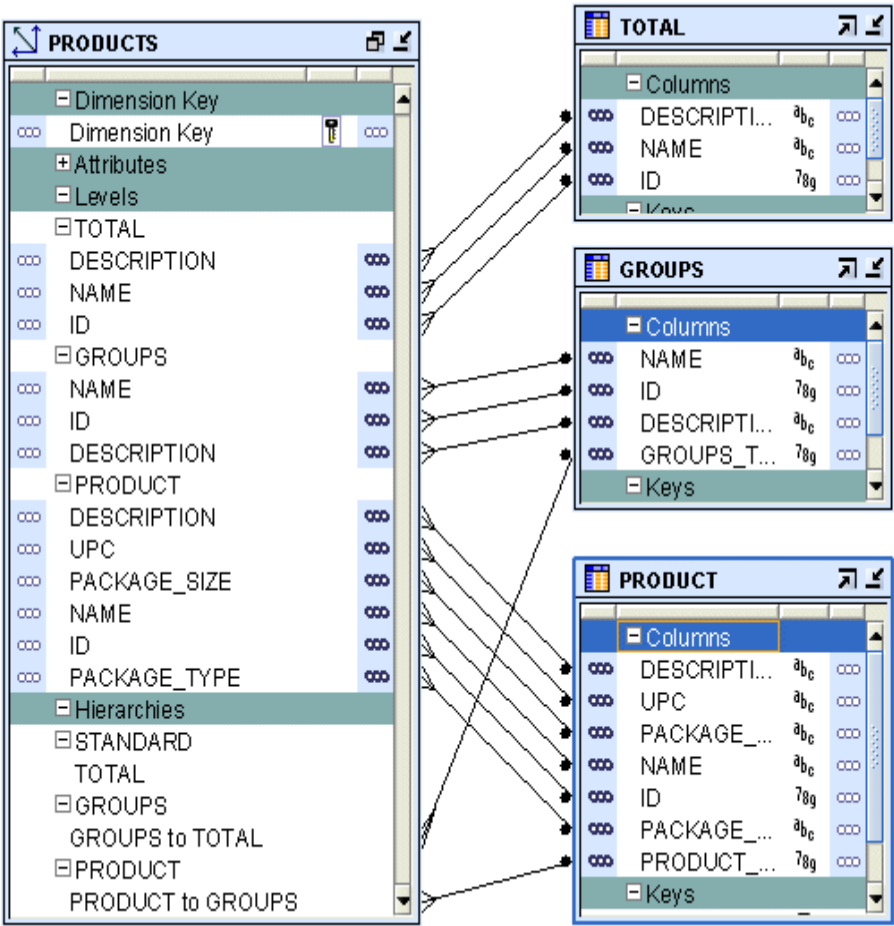
星型（star）模式：将所有的level映射到一个大的维表中。

雪花(snowflake)模式：各个不同的维表可以存在于不同的schema中。

Star Schema



Snowflake Schema



7.3.2.6 Slowly Changing Dimensions

简称就是SCD，这个概念的意思是，在维中对数据的保留时间，目前有三种类型的保留策略，默认type1可以使用，type2 和 type3 需要单独购买license。

类型	用法	描述
Type 1	覆盖重写	记录被改变之后，数据被更新，不保留历史记录
Type 2	创建一个新版本的维记录	有多个版本的维记录，当数据被更新时，会创建一个新记录保留新值
Type 3	Creating a	只有一个版本的维记录，但是这个记录

类型	用法	描述
	current value field	里的每一个属性可以保存当前以及前一个数据

7.3.2.7 创建Dimension示例

实例演示，参考《第二部分 OLAP》的有关章节

7.3.3. 立方 (Cube)

一个立方至少包含一个度量 (measures)，度量的定义：是一些可以被检查和分析的数据，我们可以对这些数据进行各种形式的统计分析，一般也就是我们最终经常要查询的统计数据，定义为度量以便进行预统计来提高数据的检索速度。

一个立方至少参考一个dimention，而且是要对应的存储类型，比如立方选择的是ROLAP类型的，那么只能参考ROLAP存储类型的dimention

数据分析经常可能用到多个维进行关联分析，这时，为了提高性能，就可以使用到立方，立方将维和度量做了初始化的聚合，所以在性能上明显高于维的操作。聚合方法可以预先定义，支持的聚合方法有

SUM, SSUM (scaled SUM), AVERAGE,
HAVERAGE (hierarchical average), MAX,
MIN, FIRST, LAST, AND, OR,
HIERARCHICAL_FIRST, HIERARCHICAL_LAST

7.3.3.1 立方的实现

有四种方式可以进行部署实现物理存储

Deployment 选项：用这个选项来指定维应该被部署到哪里：

1. Deploy Aggregations：发布这个立方度量聚合的定义。

- 2. Deploy All:** 对于一个relational implementation,立方被发布到数据库, 另外发布一个CWM语言定义到OLAP catalog。对于一个MOLAP implementation, 立方被发布到分析空间 (analytic workspace)
- 3. Deploy Data Objects only:** 只发布这个立方到数据库, 比较适合 relational implementation
- 4. Deploy to Catalog only:** 发布一个CWM 语言定义到OLAP catalog

7.3.3.2 创建Cube示例

实例演示, 参考《第二部分 OLAP》的有关章节

7.3.4. 创建映射 (mappings)

当准备好了source module之后, 我们可以开始对源数据进行 extraction, transformation, and loading (ETL)操作了, 来把源数据转换聚合等等, 最终实现在目标数据库中, 也就是target module。

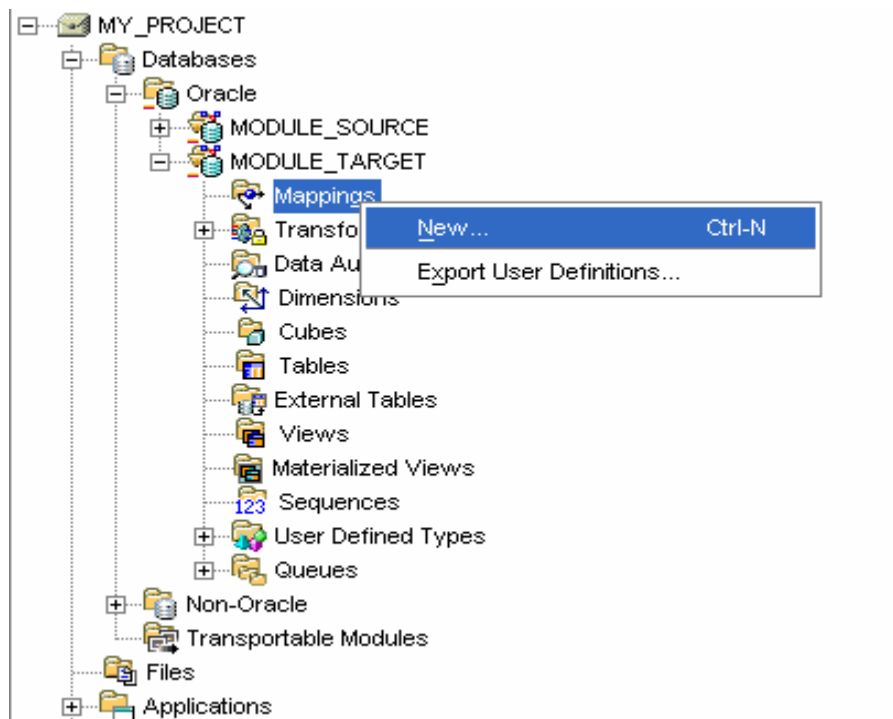
根据我们设计的mappings的种类, WB会生成相应的代码来实现数据的转换, Warehouse Builder可以为以下几种语言产生相应的代码:

- **PL/SQL:** PL/SQL 标准语言。
- **SQL*Loader:** SQL*Loader 导入平面文件的内容。
- **ABAP:** 为 SAP R/3 或者业务应用子系统而开发应用的编程语言。

定义mappings的基本步骤:

1. 创建一个Mapping
2. 添加操作符 (Operators)
3. 编辑操作符 Operators)
4. 连接操作符 (Operators)
5. Using Pluggable Mappings (可选)
6. 设置Mapping选项

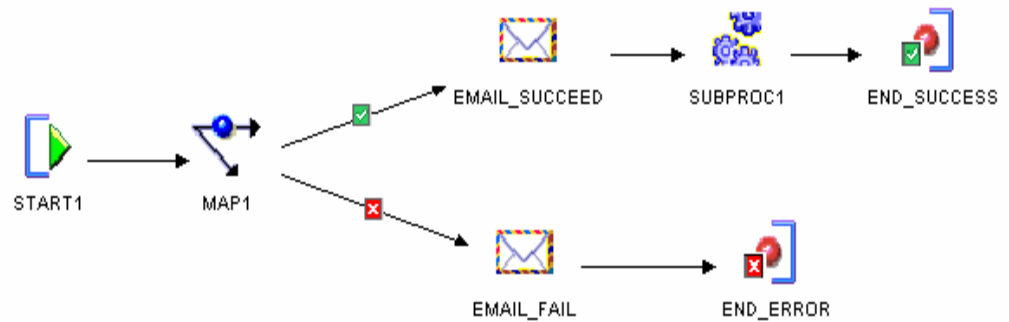
7. 设置操作符Operator, 组 Group以及 属性的选项
8. 配置Mappings Reference
9. 或者设计PL/SQL类型的mapping.
10. 调试Mapping。
11. 最后生成代码。



7.3.5. 定义过程流 (Process Flows) 可选

设计好了一个mapping以后，我们可以选择定义一个Process Flows来关联mapping和外部过程，比如电子邮件，或者Ftp指令，或者操作系统可执行脚本等等。

简单示例：



定义过程流的基本步骤：

1. 创建一个Process Flow Modules
2. 创建Process Flow Packages
3. 创建Process Flows
4. 添加行为动作
5. 连接行为动作
6. 使用参数和变量
7. 配置Process Flows
8. 校验和生成Process Flows.
9. 定时计划Process Flows(可选)
10. 发布Process Flows

7.3.6. 转换 (Transformation)

有两种方式的转换：

7.3.6.1 预定义的转换

预定义的转换分为很多类别：

◆ Administration

- ◆ Character
- ◆ Control Center
- ◆ Conversion
- ◆ Date
- ◆ Numeric
- ◆ OLAP
- ◆ Other
- ◆ Streams
- ◆ XML

7.3.6.2 自定义的转换

- ◆ 函数
- ◆ 过程
- ◆ 包

7.4. 发布到Target Schemas (deploy)

1. 创建Target Locations, 当然这个在创建Target Module时自动会创建的。
2. 使用Design Center Project Explorer来直接发布
3. 使用 Control Center Manager来发布
4. 查看发布结果信息, 确保发布准确无误
5. 发布其他Locations
6. 启动ETL过程
7. 定义ETL的作业

第二部分 OnLine Analytical Processing

第1章 OLAP概述

从以下几个方面对OLAP作一下简单的介绍

1.1. 拓扑

数据库仓库应用中，往往需要维护构建一套多维的数据库系统，以便在其中维护由标准数据库得了的多维元数据，需要特定的接口，应用才可以访问多维数据库数据

Oracle数据库多维技术的融合，在oracle9i的时候就已经开始了，在数据仓库的应用中，不再需要单独的多维在线分析数据库，通过整合，oracle提供了多维分析能力。

在整合了OLAP引擎的oracle数据库里，基于SQL的应用，就可以使用简单的SQL语句访问丰富信息内容的多维数据。

1.2. 解决的问题

举几个简单的例子：

某种产品的销售情况与去年相比有什么不同？

我们下个季度的销售应该是什么情况？

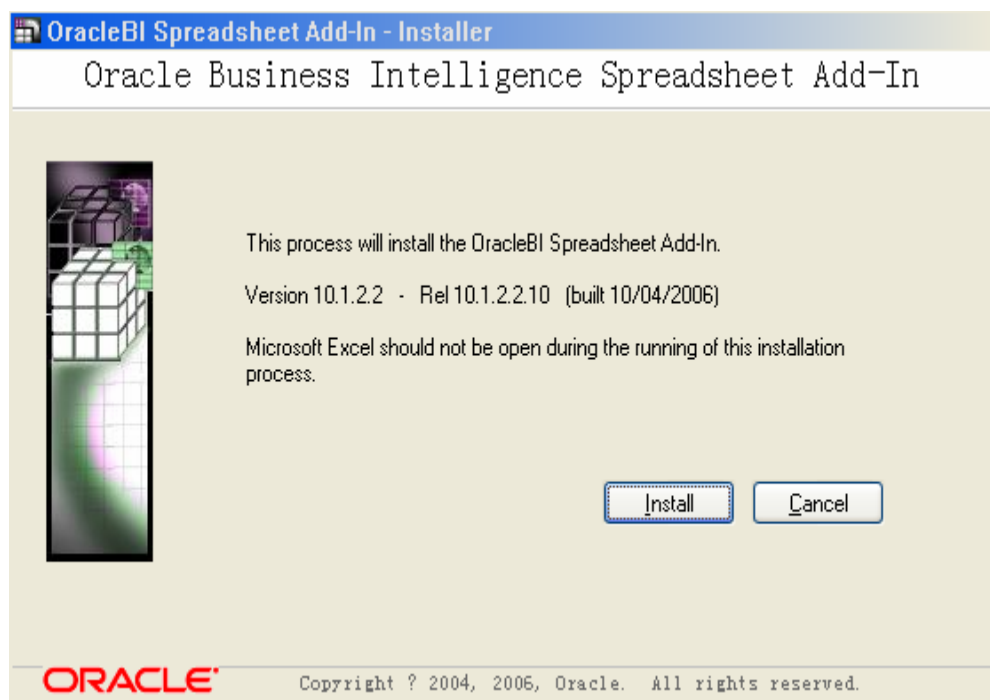
那些因素会影响我们的销售？

如果我们更改了某一个元素，会发生什么情况？

1.3. 访问工具

有两类工具可以访问OLAP数据模型：

1. OracleBI Discoverer Plus OLAP
2. OracleBI Spreadsheet Add-In



1.4. OLAP组件

OLAP Analytic Engine

集成引擎，提供多维数据支持

Analytic Workspaces

以一个多维的格式存储数据，每一个分析空间作为一个LOB存在

OLAP DML

是操纵OLAP对象的数据操纵语言，可以按照SQL DML来理解。

SQL Interface to OLAP

访问OLAP对象的SQL接口，主要是一系列的动态视图，数据库包

Analytic Workspace Java API

Java API

OLAP API

下面这两个工具是从客户端盘上上装的：

Analytic Workspace Manager

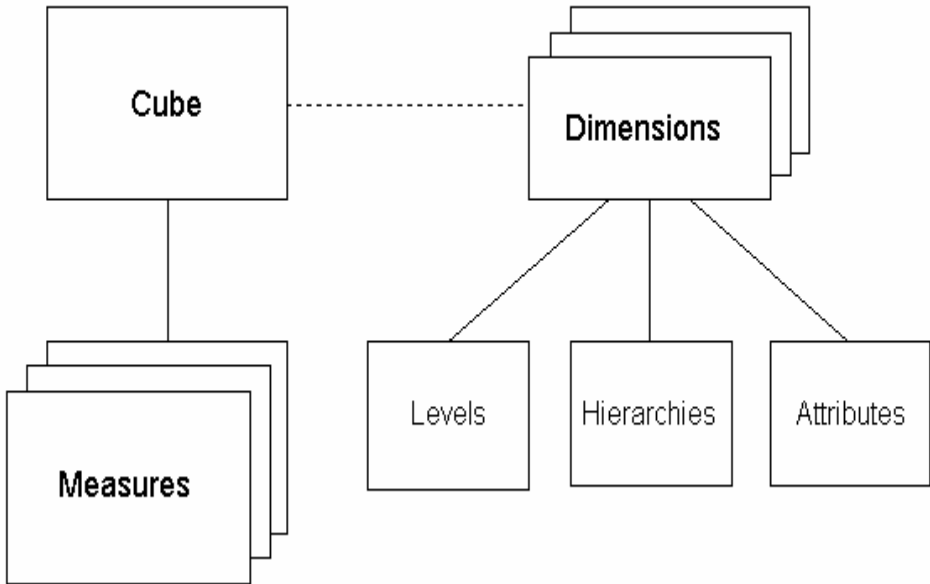
一个图形的工具，方便管理分析空间

OLAP Worksheet

类似于SQL*Plus Worksheet ， 执行OLAP DML的工具

1.5. 逻辑多维模型

Figure 2-1 Diagram of the OLAP Logical Dimensional Model



注：核心概念同OWB章节介绍的一系列多维概念

1.6.参数配置

为了最好的发挥OLAP的性能，需要对数据库的参数作一些特定的配置

JOB_QUEUE_PROCESSES	建议按如下规则设置：取值为 A + B A 代表处理器的个数，也就是 CPU 的个数 B 取值规则：每三个 CPU 增加 1
OPEN_CURSORS	>=300
PGAAggregate_TARGET	物理内存*50%

SGA_TARGET	物理内存*25%
SESSIONS	最大的并发用户数*2.5

1.7. OLAP用户

用户本身没有特殊要求，就是一个数据库用户即可，需要的额外角色不同：

OLAP用户的两个基本角色：

OLAP_USER

OLAP_DBA

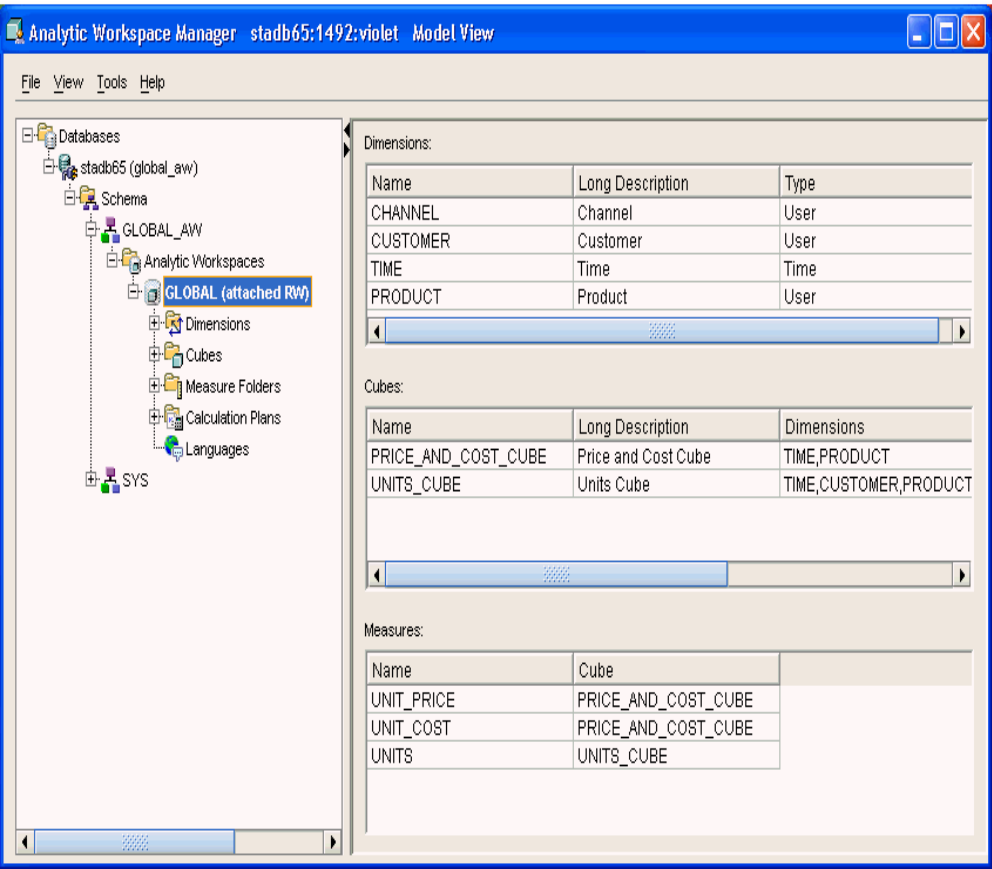
第2章 操作Analytic Workspace

2.1. Analytic Workspace Manager

管理Analytic Workspace的工具，创建，分布，修改Analytic Workspace等等。

界面分为两种视图：

Model View和Object View



Model View

一般使用，图形化来管理Analytic Workspace

Object View

高级使用，相当于一个OLAP DML的图形接口，直接修改内部的Analytic Workspace对象来实现整体的功能实现

2.2. Analytic Workspace Manager的启动

图形安装不多讲，安装完毕之后，启动Analytic Workspace Manager

Windows平台：

开始，程序， ORACLE_HOME, Integrated Management Tools, OLAP Analytic Workspace Manager and Wroksheet

Linux平台：

`$ORACLE_HOME/olap/awm/awm.sh`

最新“绿色版”的Analytic Workspace Manager，无需安装，压缩包解压之后直接可以运行。

最新版本地址：

<http://www.oracle.com/technology/products/bi/olap/index.html>

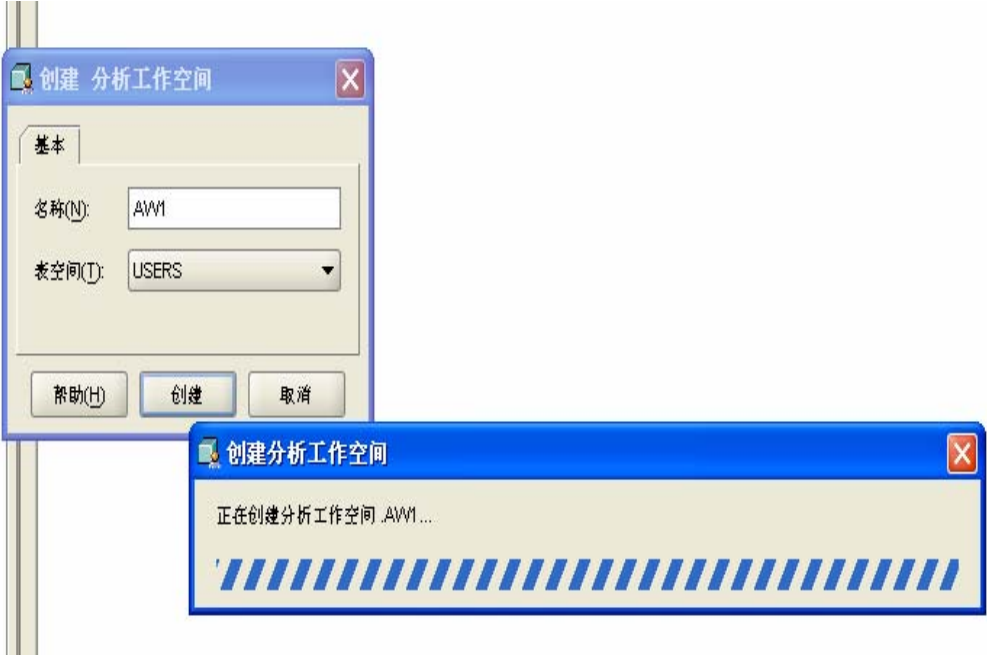
2.3. 配置环境

一般建议为OLAP应用单独创建独立的表空间，临时表空间，没有特殊要求，就是常用的数据库表空间。

2.4. 配置用户

也就是《概述》里提到的OLAP用户，此用户需要能够访问目标数据仓库用户下的必需的表，另外要赋予OLAP_USER角色

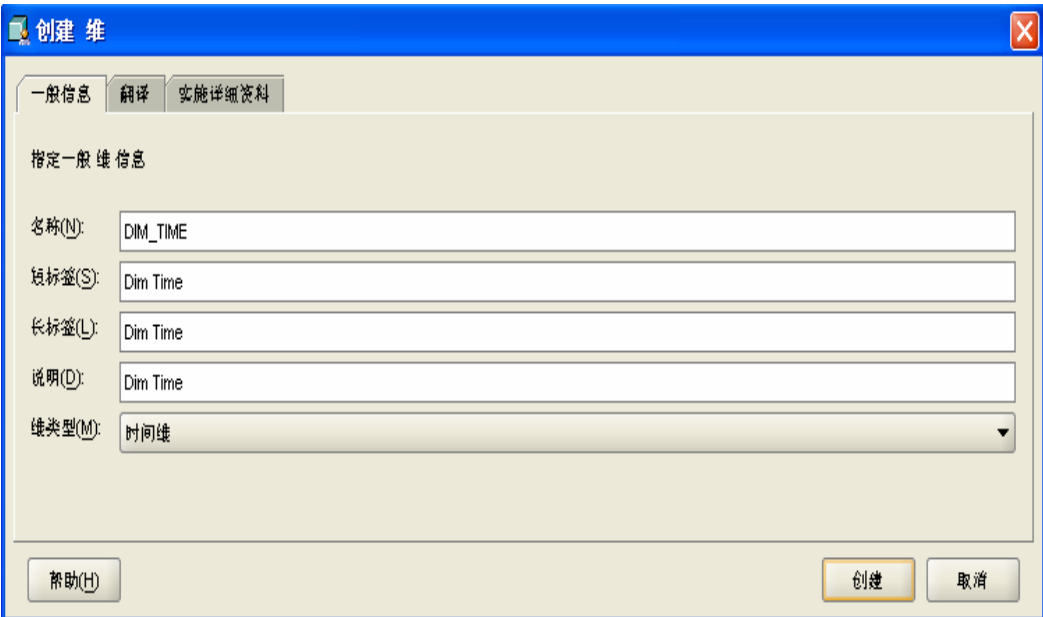
2.5. 创建Analytic Workspace





2.6. 创建逻辑Dimensions

2.6.1. 创建Dimension名称



2.6.2. 创建levels



2.6.3. 创建Hierarchies

创建 层次

一般信息

翻译

指定一般 层次 信息

名称(N):

HIER1

短标签(S):

Hier1

长标签(L):

Hier1

说明(D):

Hier1

☒ 设置为默认层次(H)

☒ 基于级别的层次(B) ☐ 基于值的层次(V)

通过从“可用”列表向“所选”列表中移动级别, 为此层次定义级别。级别在“所选”列表中的次序反映了级别在层次中的次序 (从高到低)。

可用级别(I):

>

>>

<

<<

所选级别 (从高到低)(C):

YEAR

QUARTER

MONTH

⇅

⇅

⇅

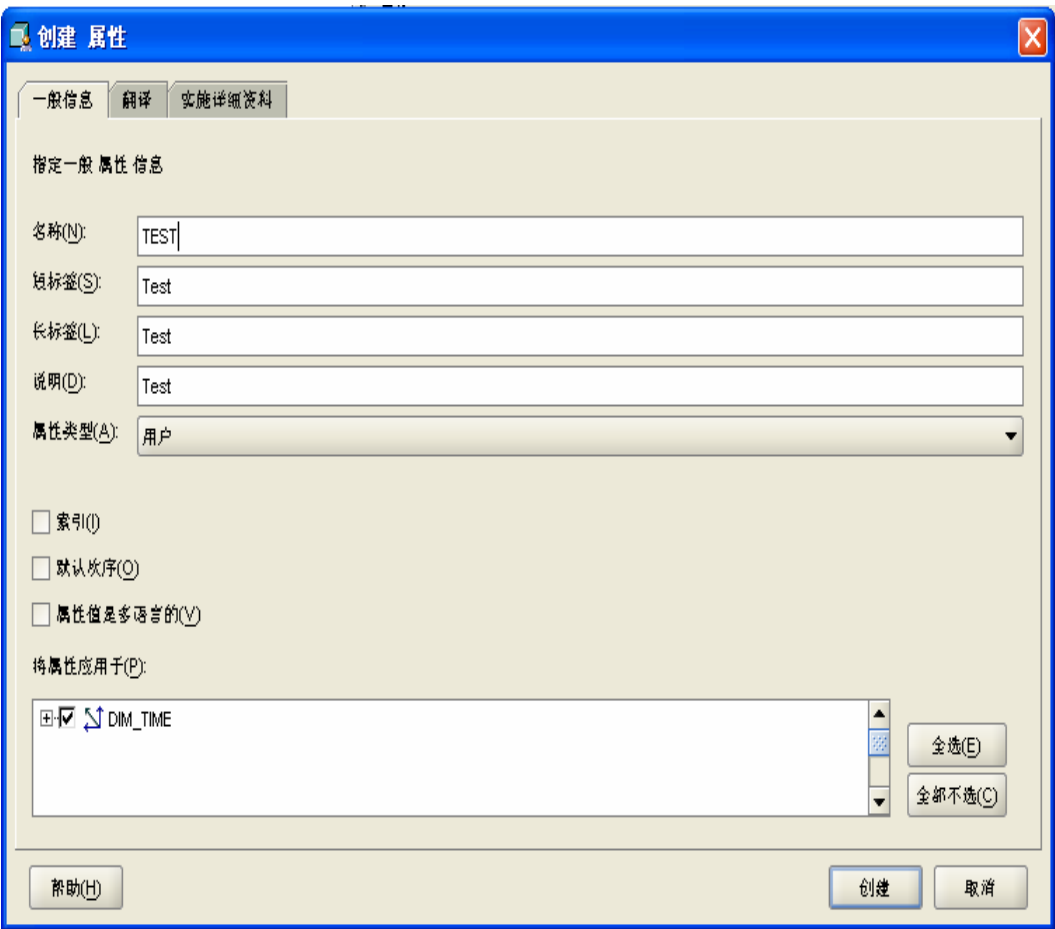
⇅

帮助(H)

创建

取消

2.6.4. 创建Attributes



2.7. 创建逻辑Cubes

2.7.1. 创建Cube名称



创建 立方

一般信息

翻译

实施详细资料

规则

汇总到

高速缓存

指定一般立方信息

名称(N): CUBE1

短标签(S): Cube1

长标签(L): Cube1

说明(D): Cube1

☒ 对立方聚集使用默认聚集计划

通过从“可用维”列表向“所选维”列表中移动维, 为此立方定义维

可用维(V):

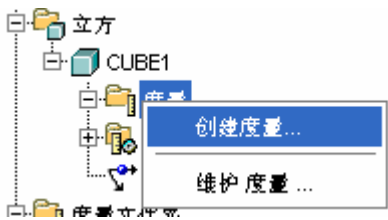
所选维(L): DIM_TIME

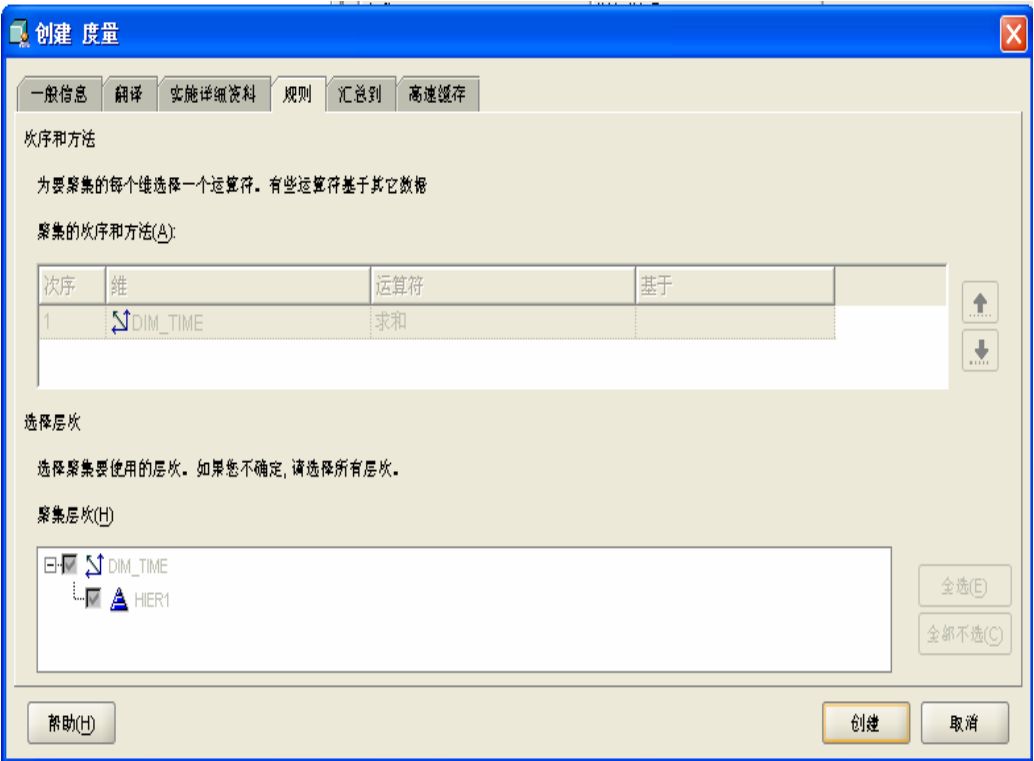
帮助(H)

创建

取消

2.7.2. 创建Measures



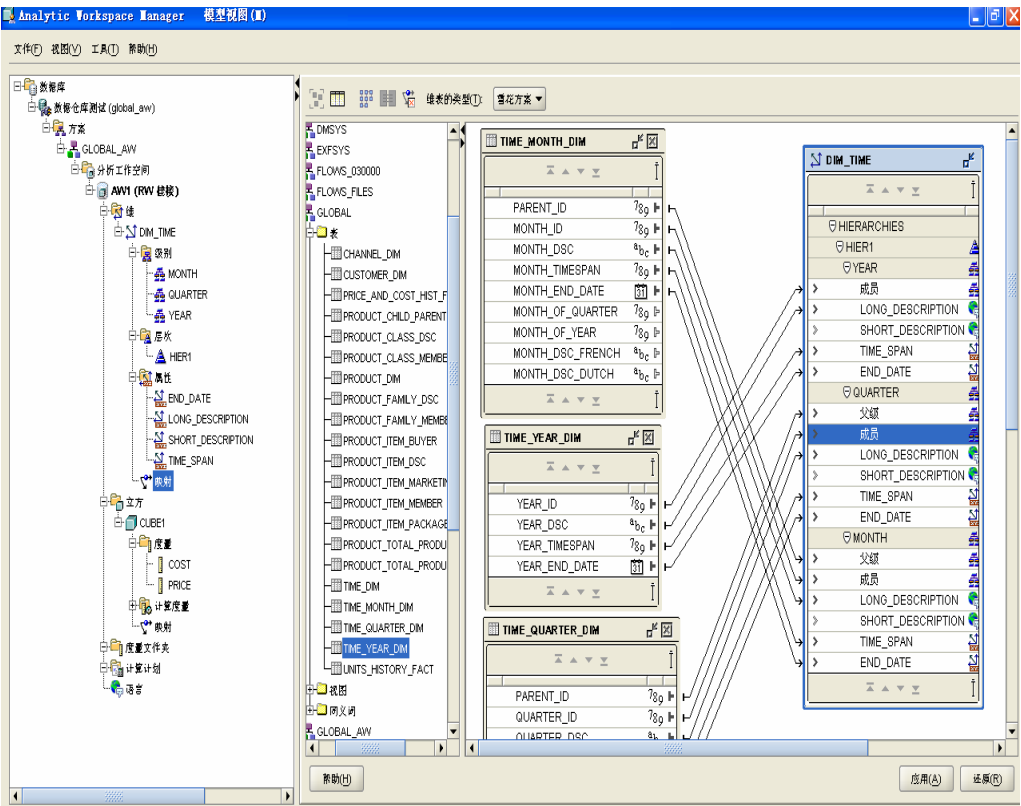


2.7.3. 创建Calculated Measures(可选)

略

2.8. 映射逻辑对象到源数据

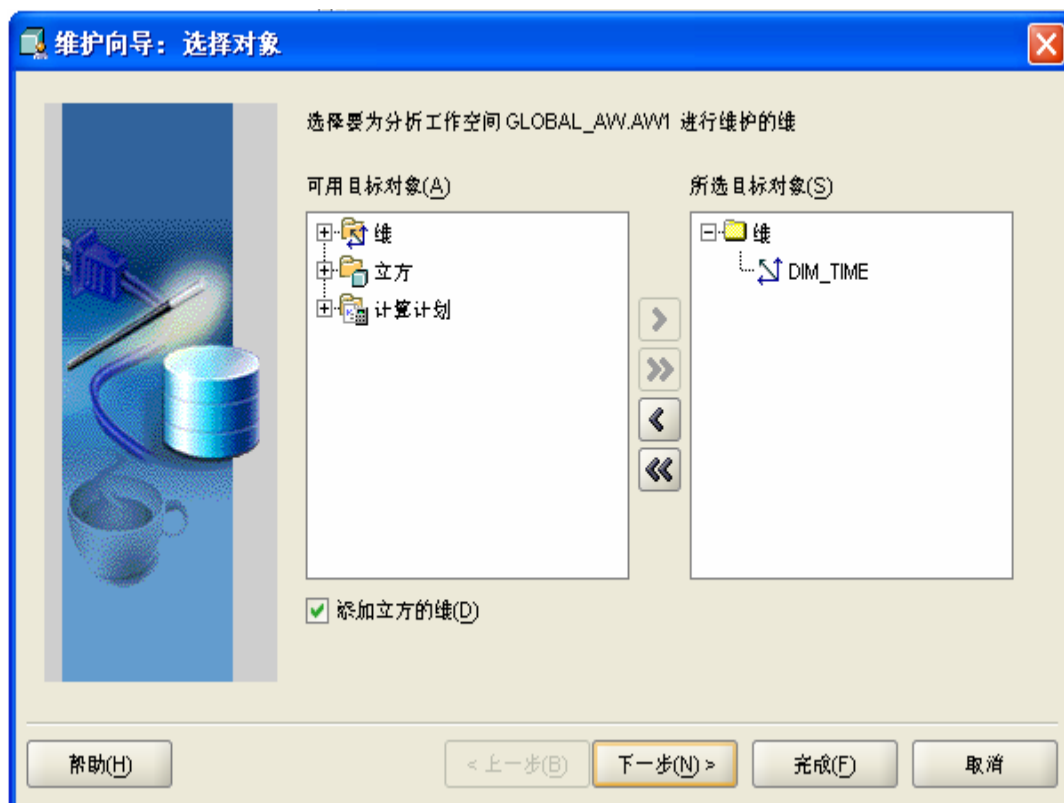
2.8.1. 映射dimension



2.8.2. 映射Cube

2.9. 加载聚合dimension和cube

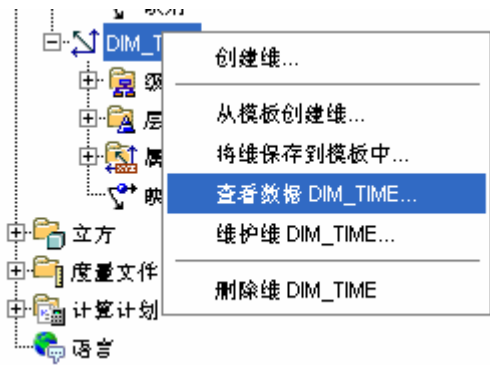


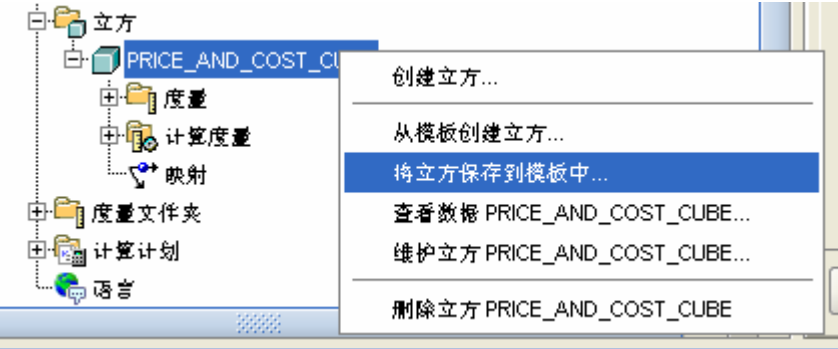


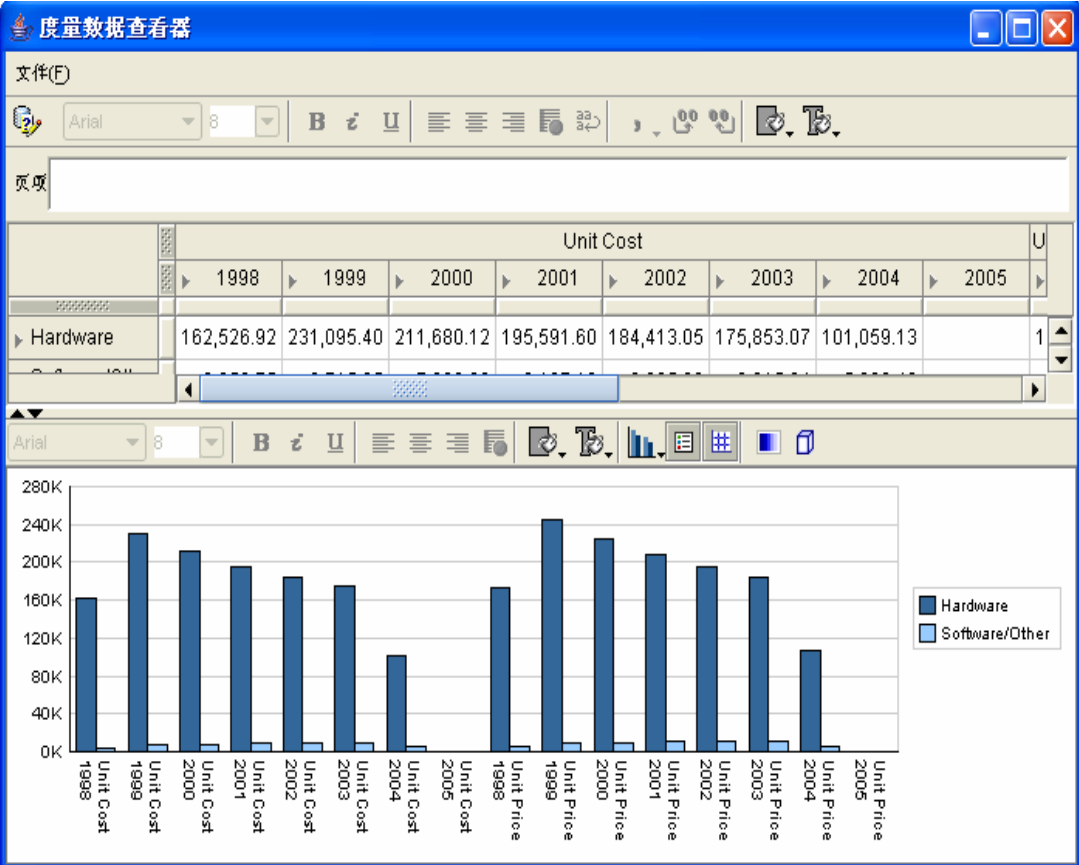
构建日志		
XML_MESSAGE	XML_AW	XML_DATE
07:53:50 Started Build(Refresh) of GLOBAL_AW.AW1 Analytic Workspa...	GLOBAL_A...	2008-01-23...
07:53:50 Attached AW GLOBAL_AW.AW1 in RVW Mode.	GLOBAL_A...	2008-01-23...
07:53:50 Started Loading Dimensions.	GLOBAL_A...	2008-01-23...
07:53:50 Started Loading Dimension Members.	GLOBAL_A...	2008-01-23...
07:53:50 Started Loading Dimension Members for DIM_TIME.DIME...	GLOBAL_A...	2008-01-23...
07:53:50 Finished Loading Members for DIM_TIME.DIMENSION. Ad...	GLOBAL_A...	2008-01-23...
07:53:50 Finished Loading Dimension Members.	GLOBAL_A...	2008-01-23...
07:53:50 Started Loading Hierarchies.	GLOBAL_A...	2008-01-23...
07:53:50 Started Loading Hierarchies for DIM_TIME.DIMENSION (1 ...	GLOBAL_A...	2008-01-23...
07:53:50 Finished Loading Hierarchies for DIM_TIME.DIMENSION. ...	GLOBAL_A...	2008-01-23...
07:53:50 Finished Loading Hierarchies.	GLOBAL_A...	2008-01-23...
07:53:50 Started Loading Attributes.	GLOBAL_A...	2008-01-23...
07:53:50 Started Loading Attributes for DIM_TIME.DIMENSION (1 ou...	GLOBAL_A...	2008-01-23...
07:53:51 Finished Loading Attributes for DIM_TIME.DIMENSION. 3 ...	GLOBAL_A...	2008-01-23...
07:53:51 Finished Loading Attributes.	GLOBAL_A...	2008-01-23...
07:53:51 Finished Loading Dimensions.	GLOBAL_A...	2008-01-23...
07:53:51 Started Updating Partitions.	GLOBAL_A...	2008-01-23...
07:53:51 Finished Updating Partitions.	GLOBAL_A...	2008-01-23...
07:53:52 Completed Build(Refresh) of GLOBAL_AW.AW1 Analytic Work...	GLOBAL_A...	2008-01-23...

2. 10. 查看dimension和cube数据

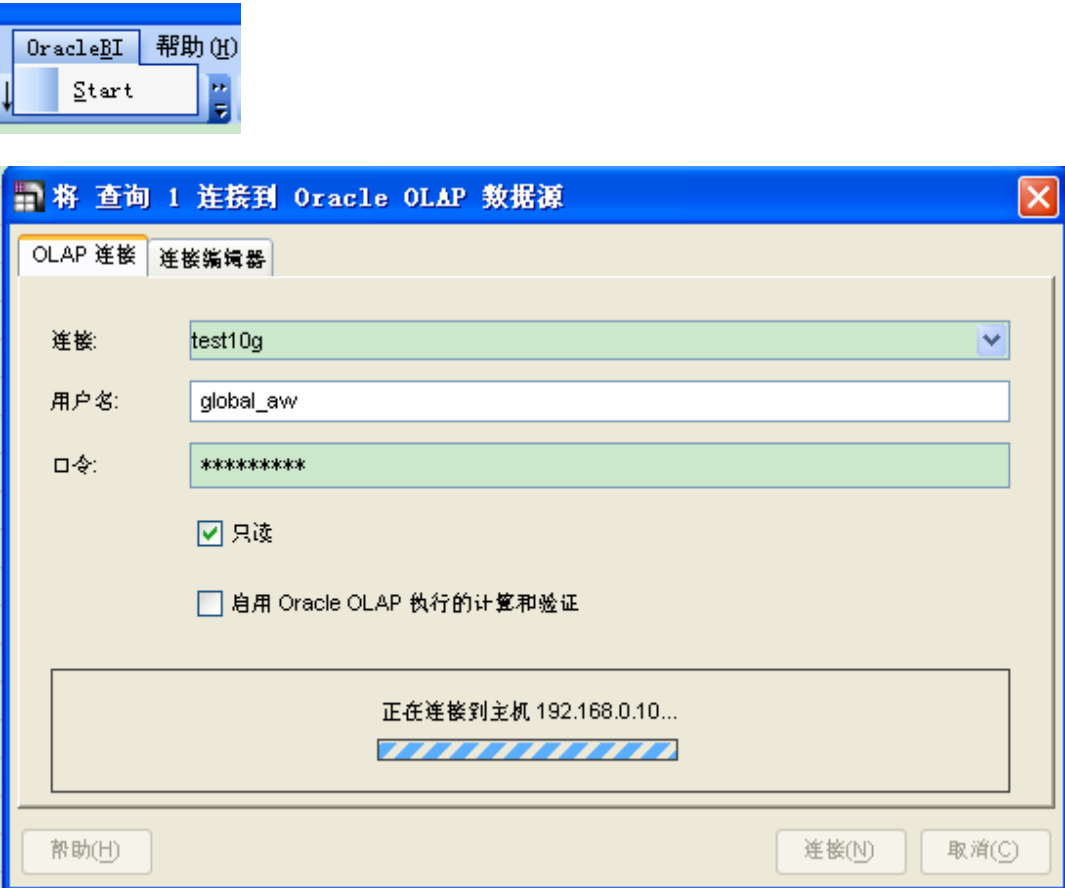
2.10.1. 使用Analytic Workspace Manager

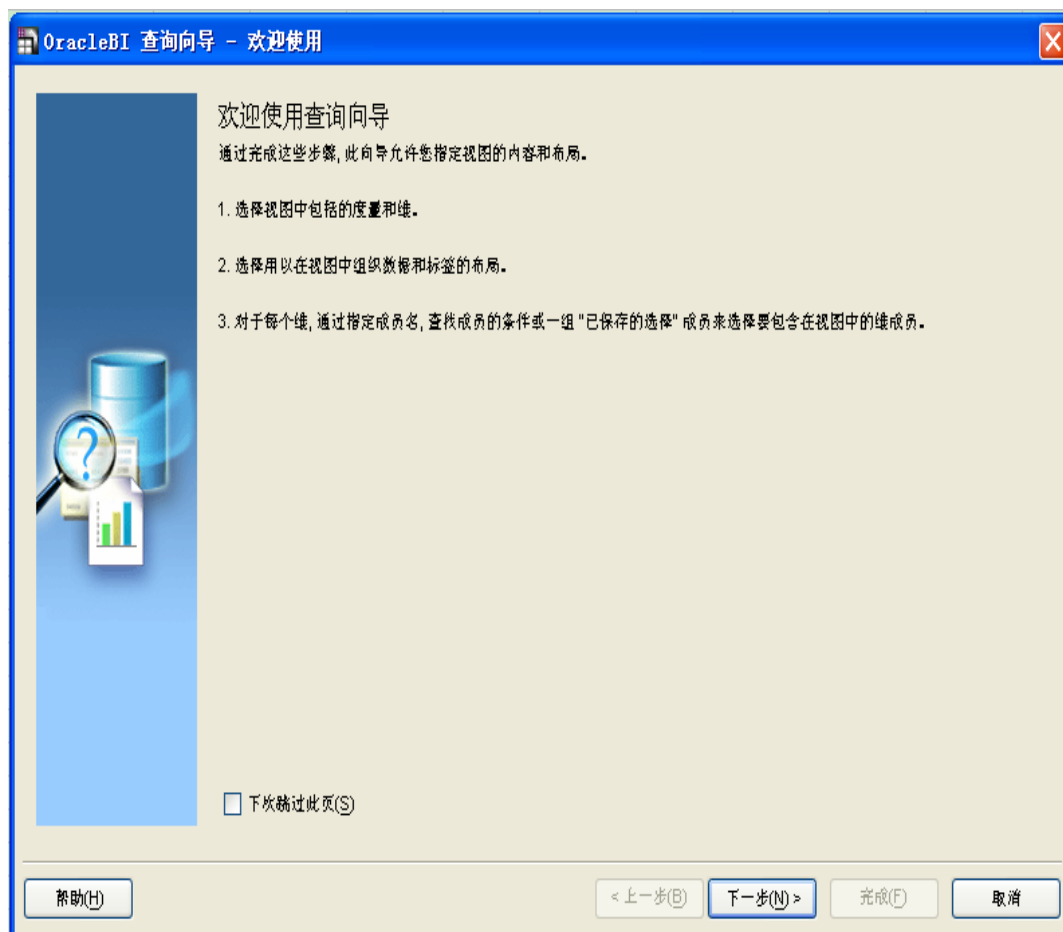


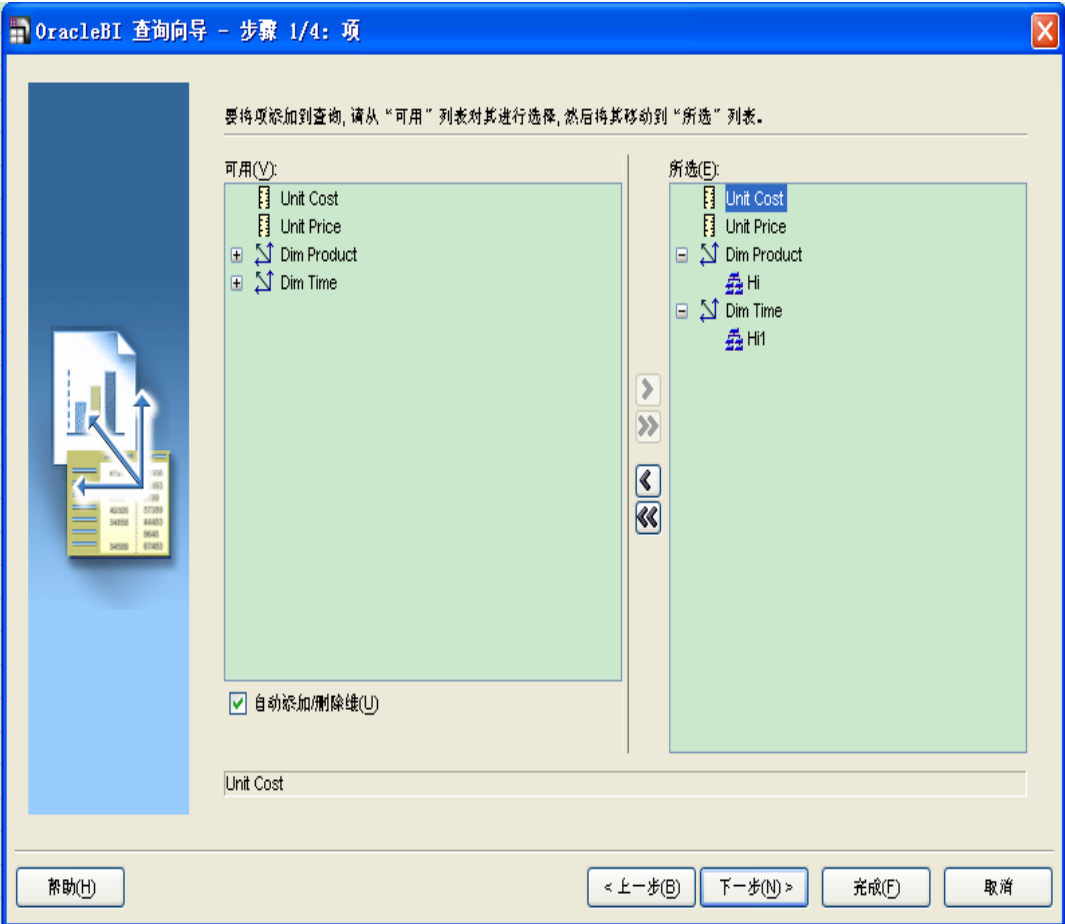




2.10.2. 使用OracleBI Spreadsheet Add-In











	A	B	C	D	E	F	G	H	I
1									
2		Unit Cost							
3		+ YEAR_1	+ YEAR_2	+ YEAR_3	+ YEAR_4	+ YEAR_85	+ YEAR_102	+ YEAR_119	+ YEAR_145
4	- Hardware	162,527	231,095	211,680	195,592	184,413	175,853	101,059	
5	+ Memory	9,782	9,597	8,723	8,836	8,730	8,832	5,162	
6	+ CD/DVD		11,727	14,828	12,982	12,113	11,584	6,682	
7	+ Portable PCs	87,870	115,406	105,481	96,980	90,912	85,890	48,921	
8	+ Desktop PCs	60,577	80,671	70,623	65,288	61,734	59,045	34,371	
9	+ Monitors	3,429	9,568	8,331	8,294	8,115	8,043	4,542	
10	+ Modems/Fax	868	4,126	3,693	3,212	2,809	2,459	1,381	
11	- Software/Other	3,957	6,715	7,894	9,187	9,095	9,015	5,220	
12	+ Documentation	650	3,419	4,516	4,416	4,350	4,282	2,472	
13	+ Operating Syst	554	549	788	1,265	1,276	1,261	733	
14	+ Accessories	2,753	2,747	2,590	3,505	3,470	3,473	2,015	

第三部分 Data Mining

第1章 ODM概述

首先介绍几个概念，逐步了解到底什么是oracle的 DM。

1.1. DataMining 函数

一种分类方法：

supervised (directed) : 去预言一个值

unsupervised (undirected) : 去发现找到数据之间的内在结构关系

另一种分类方法：

predictive : 预言性质的

descriptive : 描述性质的

1.2. Data Mining的数据准备

在使用ODM之前，我们需要准备相应的数据信息，下面介绍一系列的名词：

data set : 用于ODM分析操作的数据集合

physical organization : 表的列名

logical interpretation : 数据的逻辑解释

cases 或者 records或者examples: 数据表的一行

attributes 或者 fields : 表的一列

attributes有两种：

1. categorical : 一系列离散的，无序的分类值，比如yes, no, 比如small, medium, large, extra large

2. numerical: 大量的有序值

两种类型可以互相转换，比如年薪，可以是numerical类型的各种大小排序的值，可以转换为categorical，变成低，中，高三类值。

ODM处理的数据源必须是一个单独的数据库表或者视图，必须是一个标准的表。

支持的数据类型；

- INTEGER
- NUMBER
- FLOAT
- VARCHAR2
- CHAR
- DM_NESTED_NUMERICALS (nested column)
- DM_NESTED_CATEGORICALS (nested column)

1.3. Supervised Data Mining (predictive)

计算机算法的东西，暂时很难理解

1.4. Unsupervised Data Mining (predictive)

计算机算法的东西，暂时很难理解

1.5. Data Mining过程

Data Mining一般过程需要数据的准备，模型的创建，模型测试，计算模型，模型应用和模型部署，Oracle数据库和oracle data mining对于这些步骤提供了相应的工具

对于一个 Data Mining 的项目来说，CRISP-DM 是一个广泛使用的一个方法论。过程如下：

1. **Business Understanding:** 理解项目目标和业务前瞻性需求，然后转换这些知识为一个 data mining 问题定义和为了达到这个目标的一个初步设计。
2. **Data Understanding:** Start by collecting data, then get familiar with the data, to identify data quality problems, to

discover first insights into the data, or to detect interesting subsets to form hypotheses about hidden information.

3. **Data Preparation:** 从一系列的原始数据来构建一个最终的 **Data Set**。任务包括 table, case, attribute, transformation。

4. **Modeling:** Select and apply a variety of modelling techniques, and calibrate tool parameters to optimal values. Typically, there are several techniques for the same data mining problem type. Some techniques have specific requirements on the form of data.

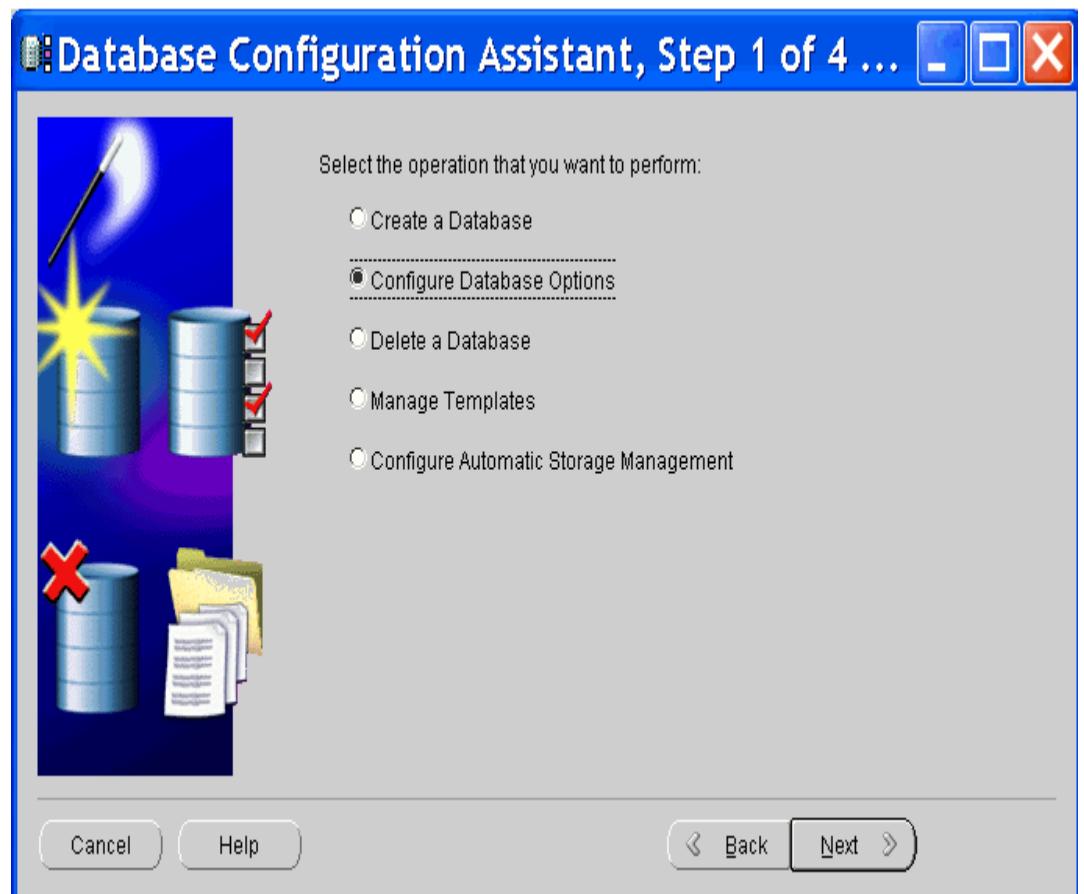
Therefore, stepping back to the data preparation phase is often needed.

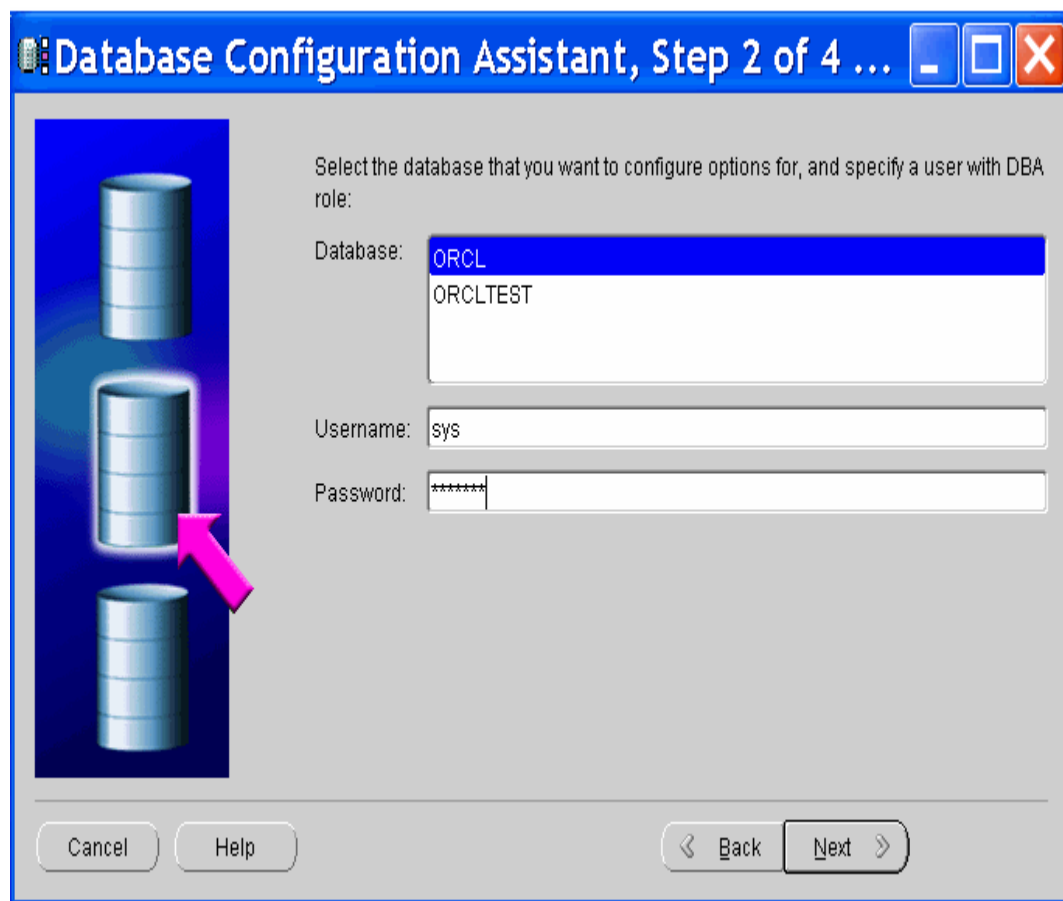
5. **Evaluation:** Thoroughly evaluate the model, and review the steps executed to construct the model, to be certain it properly achieves the business objectives. Determine if there is some important business issue that has not been sufficiently considered. At the end of this phase, a decision on the use of the data mining results is reached.

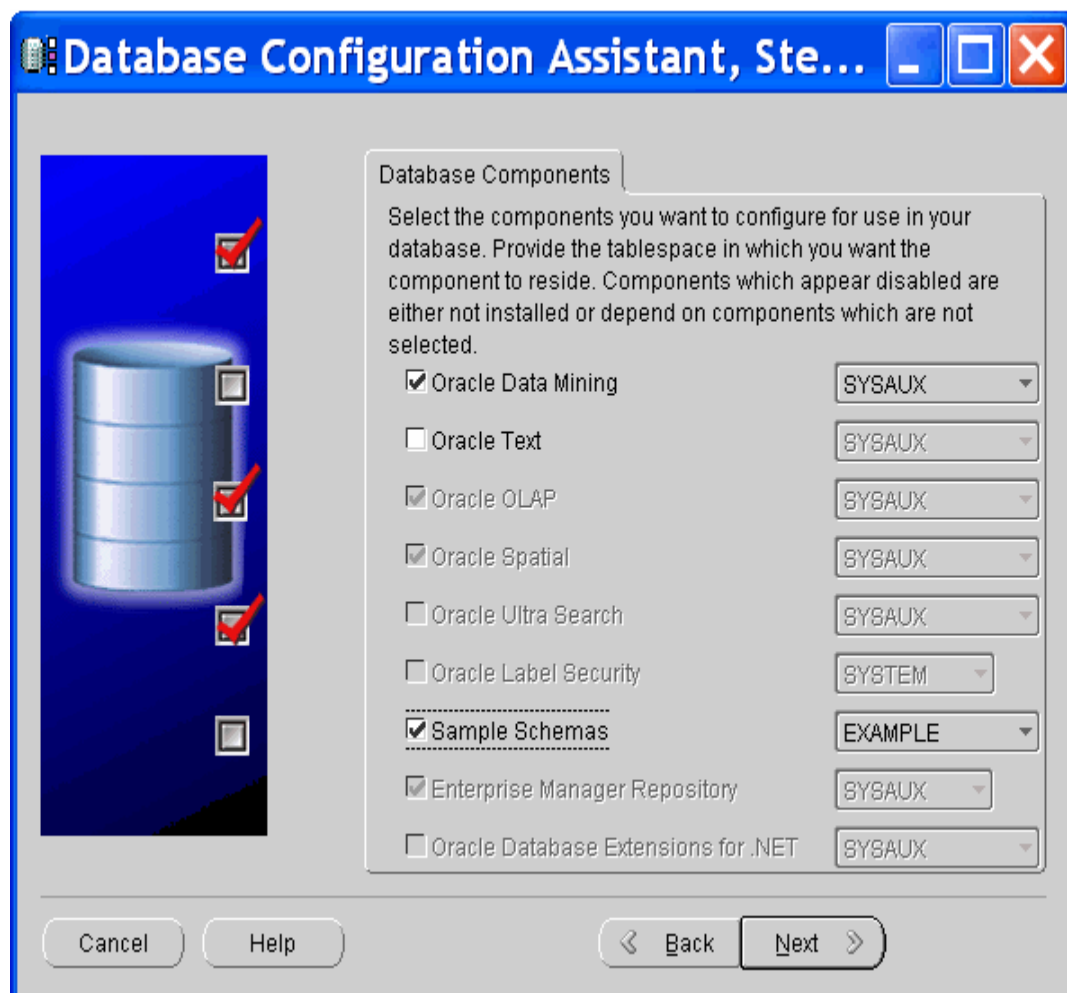
6. **Deployment:** Organize and present the results of data mining. Deployment can be as simple as generating a report or as complex as implementing a repeatable data mining process.

第2章 ODM的安装

2.1. 添加Data Mining option到数据库







安装完毕之后，可以查看是否安装成功

```
SELECT *
FROM V$OPTION
WHERE PARAMETER = 'Data Mining';
```

PARAMETER	VALUE
Data Mining	TRUE

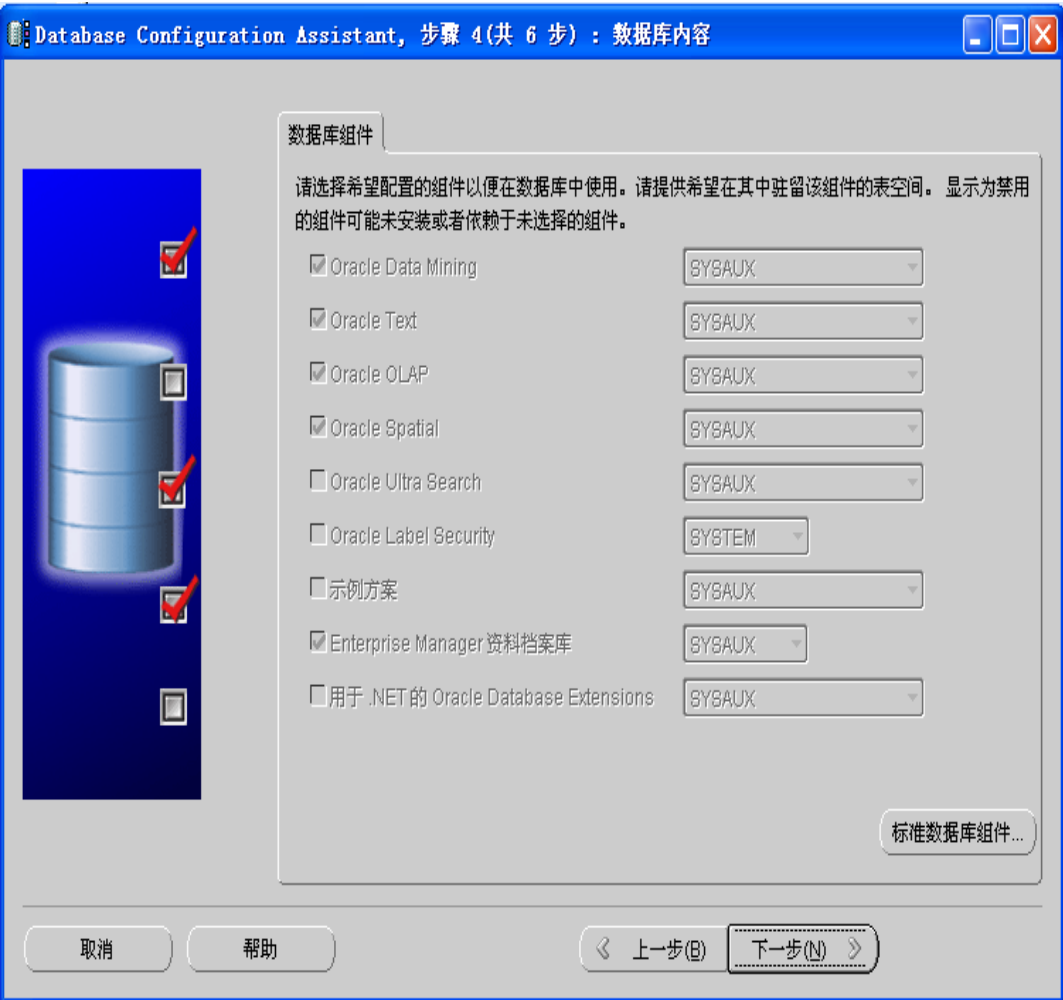
更详细的信息：

```
SELECT COMP_ID, VERSION, STATUS
FROM DBA_REGISTRY
WHERE COMP_ID = 'ODM';
```

COMP_ID	VERSION	STATUS
ODM	10.2.0.1.0	VALID

2.2. 卸载Data Mining option

卸载Data Mining选项的方法，不能通过DBCA来卸载，如下图所示，我们添加了Data Mining选项之后，再次通过DBCA来查看的话，会发现选项是不能更改的（灰色的）



卸载Data Mining的方法是：使用Universal Installer -> Deinstall Products -> Oracle Data Mining RDBMS Files -> Remove

2.3. 升级Data Mining option

升级Data Mining的方法：

普通的数据库在升级过程中，运行几个升级脚本，或者直接使用DBUA图形工具很简单的就可以进行数据库的升级，但是在升级完毕数据库之后，一定要仔细阅读关于升级补丁的相关readme.html文件，这个里面在升级完毕数据库之后，会提到很多关键性的组件选项（例如LDAP, ODM等等），需要执行特殊的升级步骤，切忌不要遗漏。

其中典型的一个就是，当数据库中有Data Mining选项时，需要执行一些后续的步骤：

2.3.1. 删除旧版本的DM对象

```
SQL> odmu101s.sql
```

2.3.2. 重建必要的DM视图

```
SQL> odm101a.sql
```

第3章 管理DM

3.1. Microsoft Windows的本地管理

因为ODM完全整合到oracle database里，所以可以使用oracle database的工具来管理ODM

所以，管理工具简单举几个例子：

Oracle Enterprise Manager Database Control

SQL*Plus

Database Configuration Assistant

Oracle Universal Installer

3.2. Linux的本地管理

与windows上相同，只是使用命令行来启动相应的工具

3.3. 远程管理

Oracle Enterprise Manager Database Control

SQL*Plus

3.4. 创建Data Mining用户

在oracle database 10gR1 , Data Mining用户叫做DMUSER, 在安装软件时就默认创建好了, 在oracle database 10gR2, DMUSER不再默认创建, 如果需要使用DM, 需要手工创建。

例子:

1. 由于DM需要处理大量的数据和逻辑运算, 建议创建单独表空间和临时表空间

```
CREATE TABLESPACE ODMPerm DATAFILE.....;
```

```
CREATE TEMPORARY TABLESPACE ODMTEMP TEMPFILE.....;
```

2. 创建用户

```
CREATE USER dmuser1 IDENTIFIED BY .....
```

3. 必须拥有的权限

```
GRANT create procedure to DMUSER1
```

```
GRANT create session to DMUSER1
```

```
GRANT create table to DMUSER1
```

```

GRANT create sequence to DMUSER1

GRANT create view to DMUSER1

GRANT create job to DMUSER1

GRANT create type to DMUSER1

GRANT create synonym to DMUSER1

GRANT execute on ctxsys.ctx_ddl to DMUSER1

```

DMSYS用户是拥有的是所有DM用户创建的DM模型数据，它的默认表空间是SYS_AUX，若使用大量的操作，为了不使得SYS_AUX异常增长，可以考虑将DM的默认表空间迁移出来。

注：顺便简单介绍一下V\$SYS_AUX_OCCUPANTS视图。

3.5. 导出导入Data Mining模型

很明显，DM数据是在DATABASE里的，所以可以使用expdp和impdp来进行DM数据的导出导入，支持Database level和schema level。

另外，有一种单独针对DM模型进行导出导入的方法，而且这些方法可以直接从expdp的dump文件中直接提取模型来进行导入。

```

EXECUTE
DBMS_DATA_MINING.EXPORT_MODEL('allmodels.dmp','DMTEST');

EXECUTE
DBMS_DATA_MINING.IMPORT_MODEL('allmodels01.dmp','DMTEST');

```

其中DMTEST是一个directory

注：exp和imp不支持这种操作。

第4章 DM工具安装及使用

Oracle提供了两种辅助分析工具

1. Oracle Data Miner
2. Oracle Spreadsheet Add-In for Predictive Analytics

4.1. Oracle Data Miner

目前Oracle Data Miner 支持 Windows 2000, Windows XP Professional Edition 和 Linux.

安装步骤:

For windows

1. 下载压缩包

<http://www.oracle.com/technology/products/bi/odm/index.html>.

2. 解压.

3. 呵呵，绿色的，解压就能用，很方便。

For linux

1. 确保java版本在1.4.2或者以上

2. 下载

3. 解压

```
unzip odminer.zip -d odminer
```

4. Grant execution permission

```
chmod +x odminer/bin/odminer
```

5. odminer

4.2. Oracle Spreadsheet Add-In for Predictive Analytics

系统需求

- Microsoft Excel 2000, 2002, or 2003
- Oracle Objects for OLE

安装步骤:

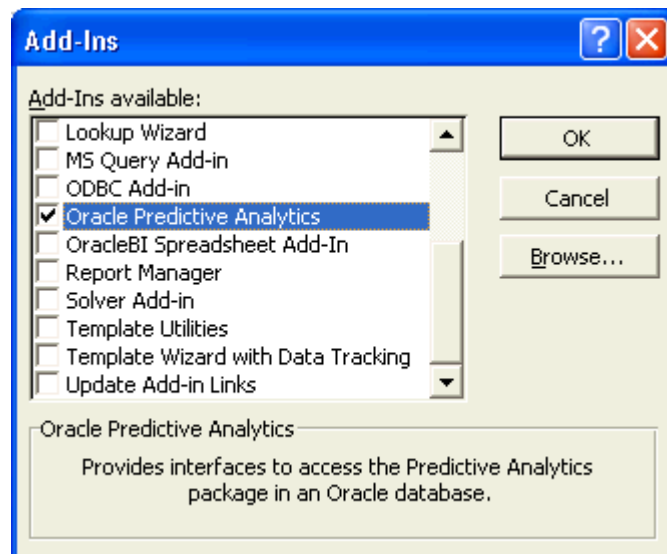
1. 下载安装包

<http://www.oracle.com/technology/products/bi/odm/index.html>

2. 解压文件 Predictive_Analytics.xla 到Microsoft Office Library目录, 库目录如下: C:\Program Files\Microsoft Office\Office\Library

3. 打开 Tools > Add-Ins.

4. 选择Oracle Predictive Analytics .



附录A OWB各种工具启动方法

OWB 工具	Windows 环境下: 选择开始→程序→ OWB_ORACLE_HOME	Linux 环境下: cd \$OWB_ORACLE_HOME/owb/bin/unix
Repository Assistant	Administration→Repository Assistant.	reposinst.sh
Design Center	Design Center.	owbclient.sh
Start Control Center Service	Administration→Start Control Center Service.	local_service_login.sh 如果启动失败的话, 可以使用 \$OWB_ORACLE_HOME/owb/rtp/sql/service_doctor.sql.
Control Center Manager	先启动 Design Center→选择 tools 菜单, 选择 Control Center Manager.	local_service_login.sh -startup <OWB-Home>
Stop Control Center Service	Administration→Stop Control Center Service.	local_service_login.sh -closedown <OWB-Home>
Start OWB Browser Listener	Administration→Start OWB Browser Listener.	startOwbbInst.sh
Repository Browser	Repository Browser.	启动 OWB Browser Listener 然后 openRAB.sh.
Stop OWB Browser Listener	Administration→Stop OWB Browser Listener.	stopOWBBInst.sh
OMB Plus	OMB Plus.	OMBPlus

附录B OLAP活动目录视图

PUBLIC Synonym	Description
ALL_OLAP2_AWS	Lists the analytic workspaces.
ALL_OLAP2_AW_ATTRIBUTES	List of dimension attributes in analytic workspaces.
ALL_OLAP2_AW_CATALOGS	Lists the measure folders in analytic workspaces.
ALL_OLAP2_AW_CATALOG_MEASURES	Lists the measures in the measure folders.
ALL_OLAP2_AW_CUBES	List of cubes in analytic workspaces.
ALL_OLAP2_AW_CUBE_AGG_LVL	List of levels in aggregation plans in analytic workspaces.
ALL_OLAP2_AW_CUBE_AGG_MEAS	List of measures in aggregation plans in analytic workspaces.
ALL_OLAP2_AW_CUBE_AGG_OP	List of aggregation operators in aggregation plans in analytic workspaces.
ALL_OLAP2_AW_CUBE_AGG_SPECS	List of aggregation plans in analytic workspaces.
ALL_OLAP2_AW_CUBE_DIM_USES	List of cubes with their associated dimensions in analytic workspaces.
ALL_OLAP2_AW_CUBE_MEASURES	List of cubes with their associated measures in analytic workspaces.
ALL_OLAP2_AW_DIMENSIONS	List of dimensions in analytic workspaces.
ALL_OLAP2_AW_DIM_HIER_LVL_ORD	List of hierarchical levels in analytic workspaces.
ALL_OLAP2_AW_DIM_LEVELS	List of levels in analytic workspaces.
ALL_OLAP2_AW_PHYS_OBJ	List of standard form objects in analytic workspaces.
ALL_OLAP2_AW_PHYS_OBJ_PROP	List of properties associated with

PUBLIC Synonym	Description
	standard form objects in analytic workspaces.

附录C OLAP动态性能视图

V\$AW_AGGREGATE_OP	Lists the aggregation operators available in the OLAP DML.
V\$AW_ALLOCATE_OP	Lists the allocation operators available in the OLAP DML.
V\$AW_CALC	Collects information about the use of cache space and the status of dynamic aggregation.
V\$AW_LONGOPS	Collects status information about SQL fetches.
V\$AW_OLAP	Collects information about the status of active analytic workspaces.
V\$AW_SESSION_INFO	Collects information about each active session.