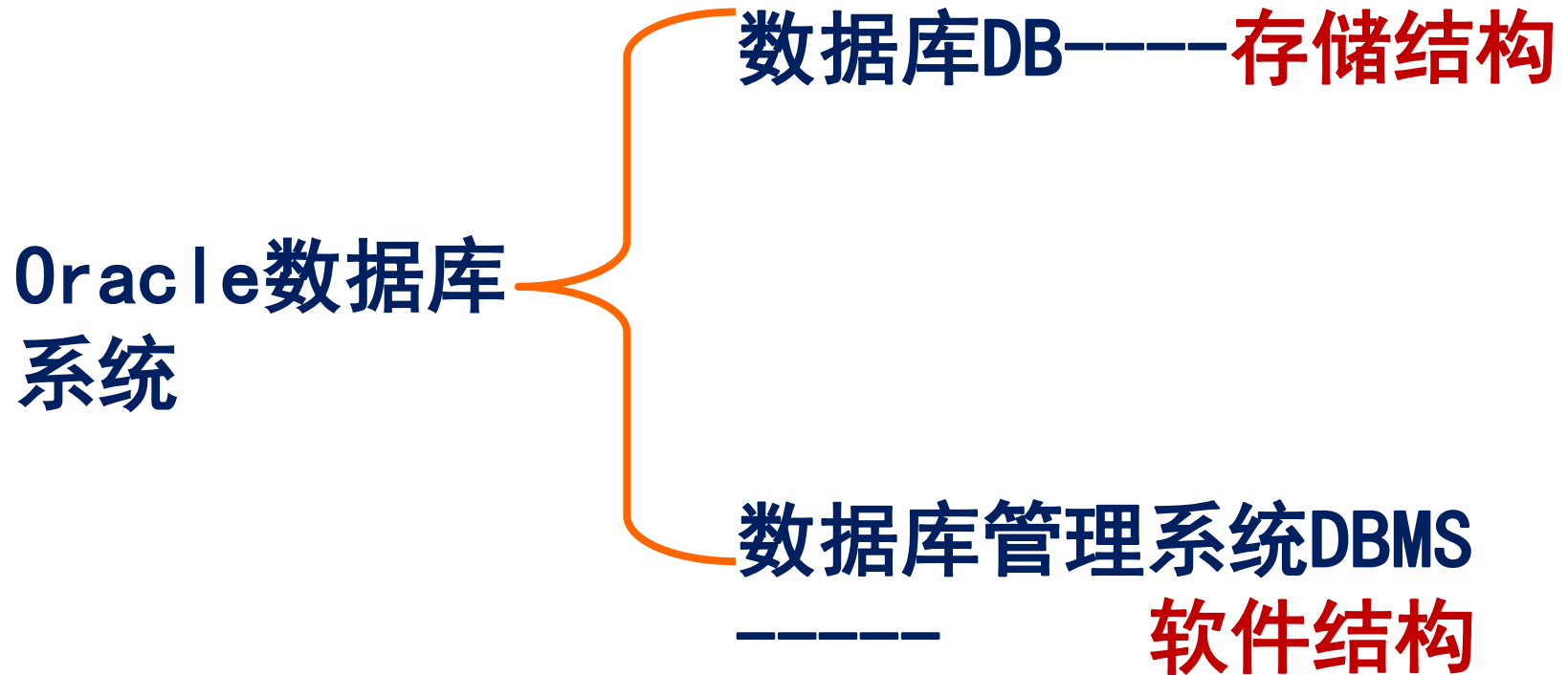




大型数据库应用技术

Oracle基本操作

授课教师：王欢



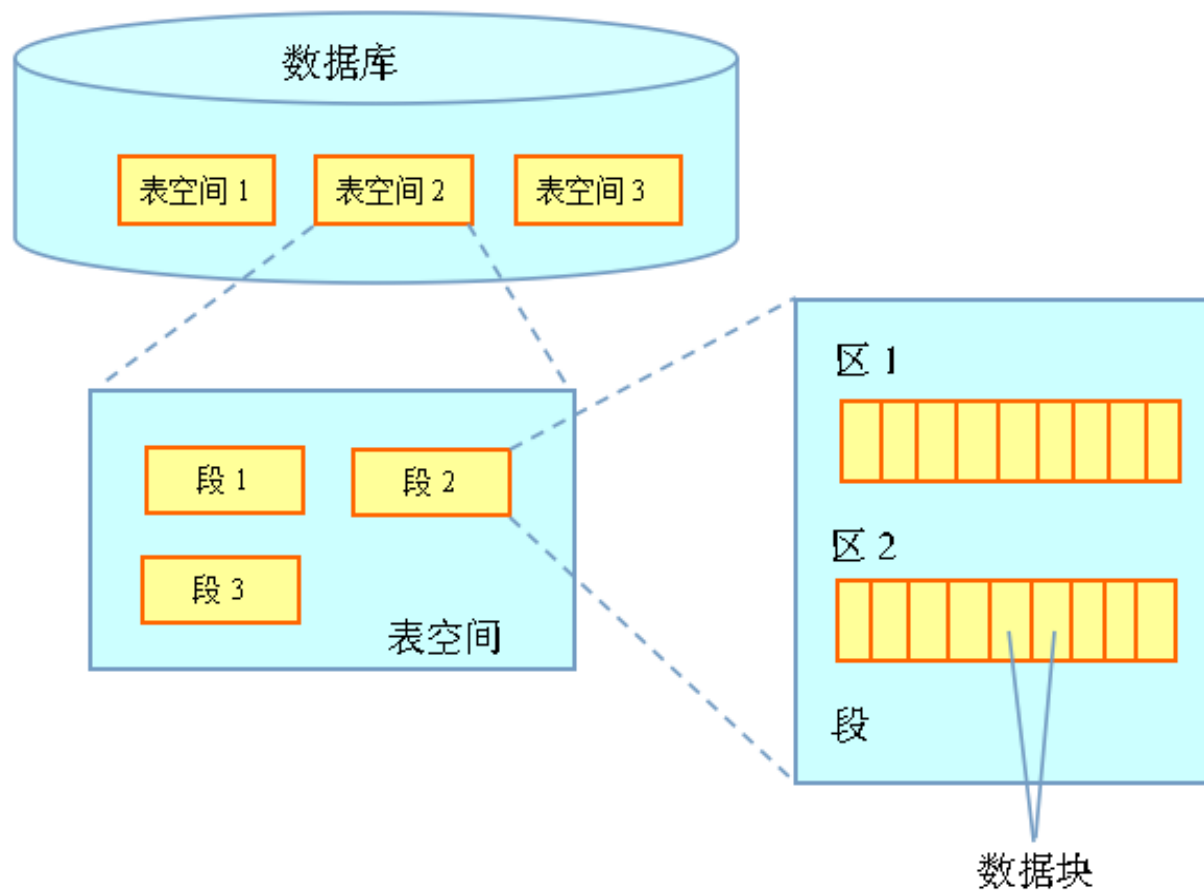


存储结构

逻辑存储结构

物理存储结构

逻辑存储结构





» 数据块 (Data Blocks)

- ★ 数据块是Oracle逻辑存储结构中的最小的逻辑单位
- ★ 一个数据库块对应一个或者多个物理块，大小由参数DB_BLOCK_SIZE决定

```
SQL> col name format a30
SQL> col value format a20
SQL> select name,value from v$parameter where name = 'db_block_size';
```

NAME	VALUE
db_block_size	8192

```
SQL> _
```

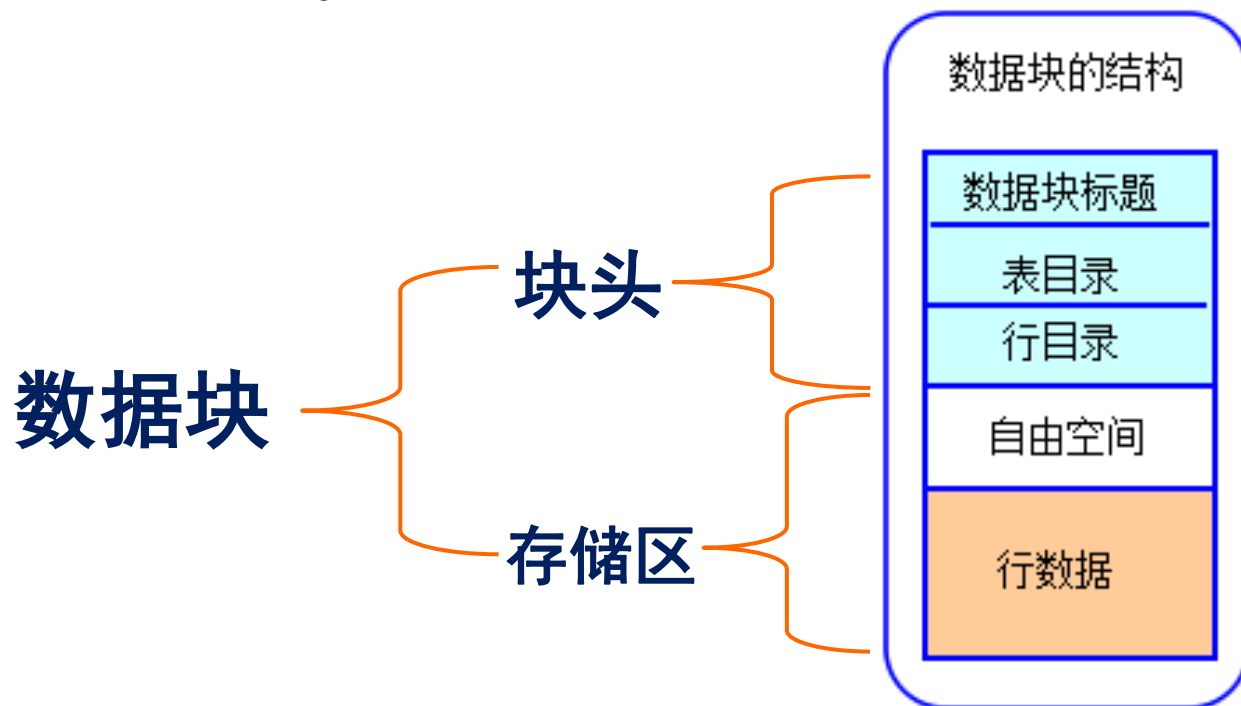


» 数据块 (Data Blocks)

- ★ 数据块是Oracle逻辑存储结构中的最小的逻辑单位
- ★ 一个数据库块对应一个或者多个物理块，大小由参数DB_BLOCK_SIZE决定
- ★ 数据块的结构包括块头和存储区的两个部分



数据块组成





» 数据区 (Extent)

♥ 数据区是由连续的数据块结合而成的

♥ 数据区是Oracle存储分配的最小单位

数据块是最小的逻辑单位 是Oracle最小的读取单位。
段空间使用完了，系统自动分配新的数据区，所以区是存储分配的最小单位，Oracle以区为单位进行存储空间的扩展。



» 段 (Segment)

- ✓ 数据段: 存储表中所有数据
- ✓ 索引段: 存储表上最佳查询的所有索引数据
- ✓ 临时段: 存储表排序操作期间建立的临时表的数据
- ✓ 回滚段: 存储修改之前的位置 and 值



»» 表空间 (TableSpace)

- 👉 表空间是数据库的最大逻辑划分区域
- 👉 一个表空间由一个或多个数据文件组成，一个数据文件只属于一个表空间
- 👉 表空间的大小是它所对应的数据文件大小的总和



» 默认创建的表空间

系统表空间 (system tablespace)

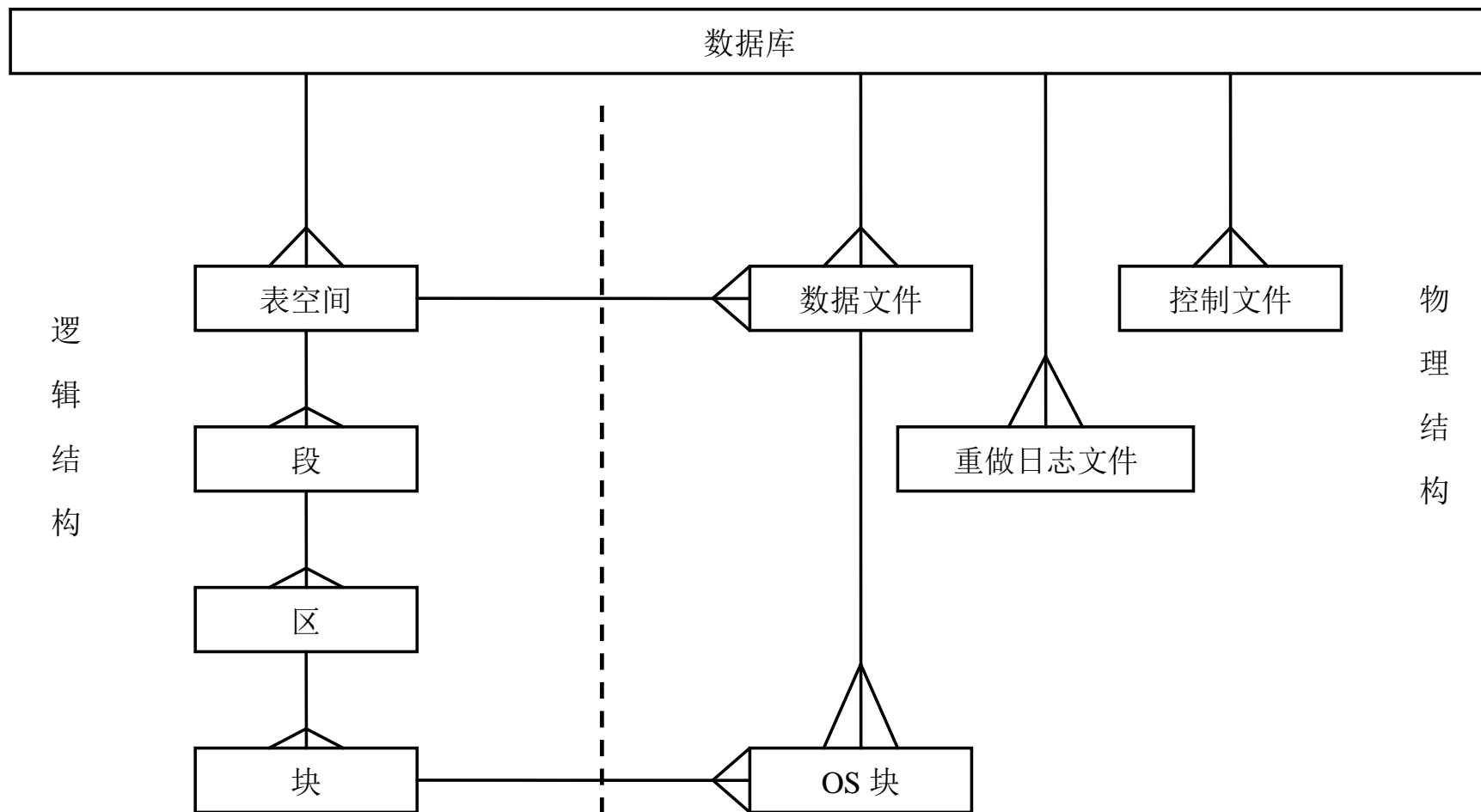
辅助表空间 (sysaux tablespace)

撤销表空间 (undo tablespace)

临时表空间 (temp tablespace)

用户表空间 (users tablespace)

复习 体系结构



物理结构

逻辑结构

体系结构



物理 存储结构

数据文件

系统数据文件

(SYSTEM01.DBF和SYSAUX01.DBF)

回滚数据文件 (UNDOTBS01.DBF)

用户数据文件

(USERS01.DBF、TBSP_1.DBF)

临时数据文件 (TEMP02.DBF)

控制文件



CONTROL
01.CTL

最重要的文件，建库之前就要在初始化参数文件中把路径设置好

日志文件

重做日志文件

记录所有数据变化，提供恢复机制

归档日志文件

重做日志的历史备份

其他文件：服务器参数文件、密码文件、警告文件（运行+错误）、跟踪文件



文件(F) 编辑(E) 查看(V) 工具(T) 帮助(H)

组织 包含到库中 共享 新建文件夹

名称 修改日期 类型 大小

名称	修改日期	类型	大小
CONTROL01.ctl	2014/1/8 11:24	Visual Basic Use...	11,984 KB
DBSQL1_1781633366_1.DBF	2014/1/8 11:24	Microsoft Visual...	2,224 KB
EXAMPLE01.DBF	2014/1/8 11:24	Microsoft Visual...	102,408 KB
REDO01.LOG	2014/1/8 11:24	文本文档	51,201 KB
REDO02.LOG	2014/1/8 11:24	文本文档	51,201 KB
REDO03.LOG	2014/1/8 11:24	文本文档	51,201 KB
SYS_AUX01.DBF	2014/1/8 11:24	Microsoft Visual...	593,928 KB

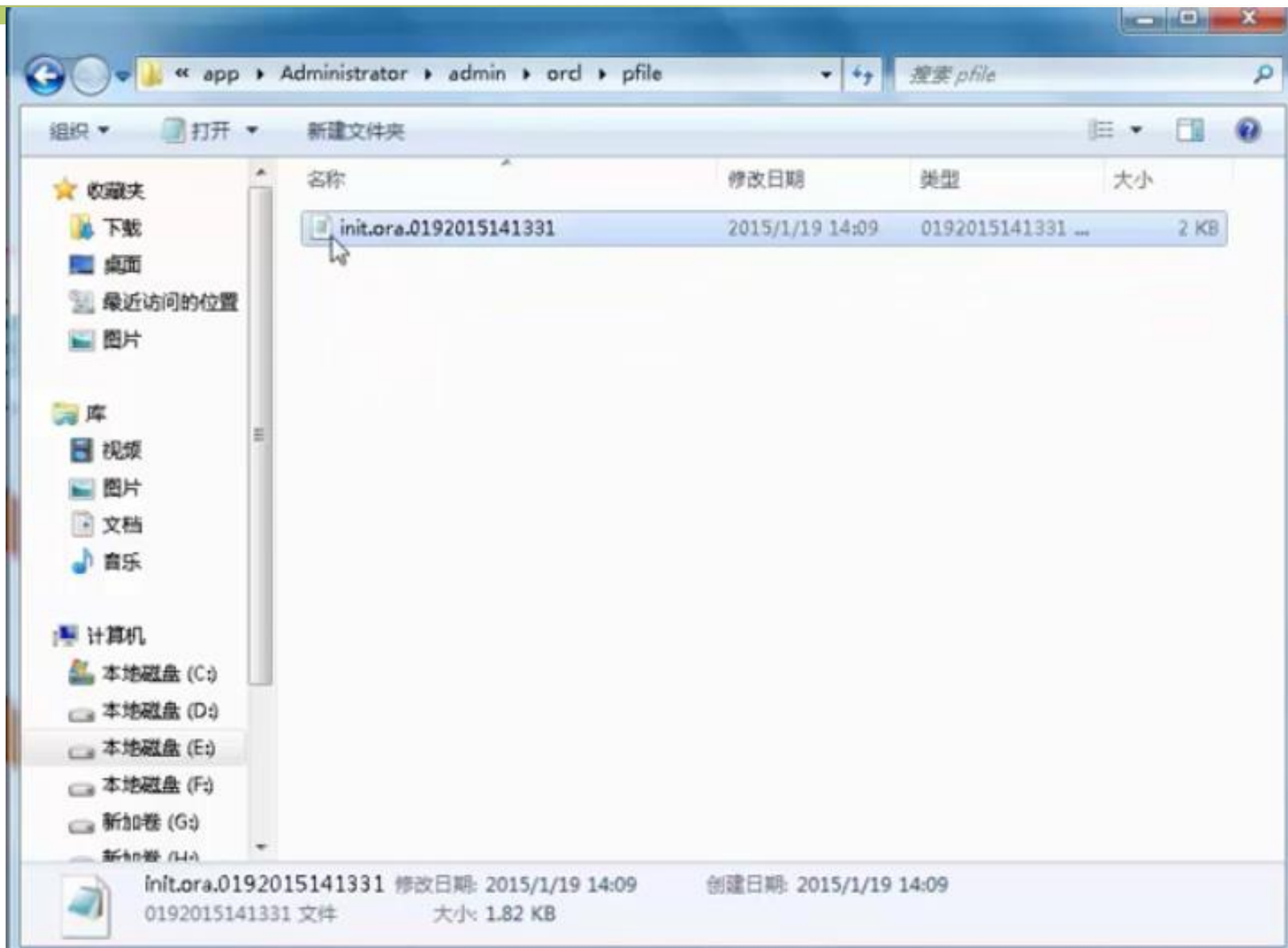
SQL> col file_name for a50
SQL> set linesize 100;
SQL> select file_name, tablespace_name from dba_data_files;

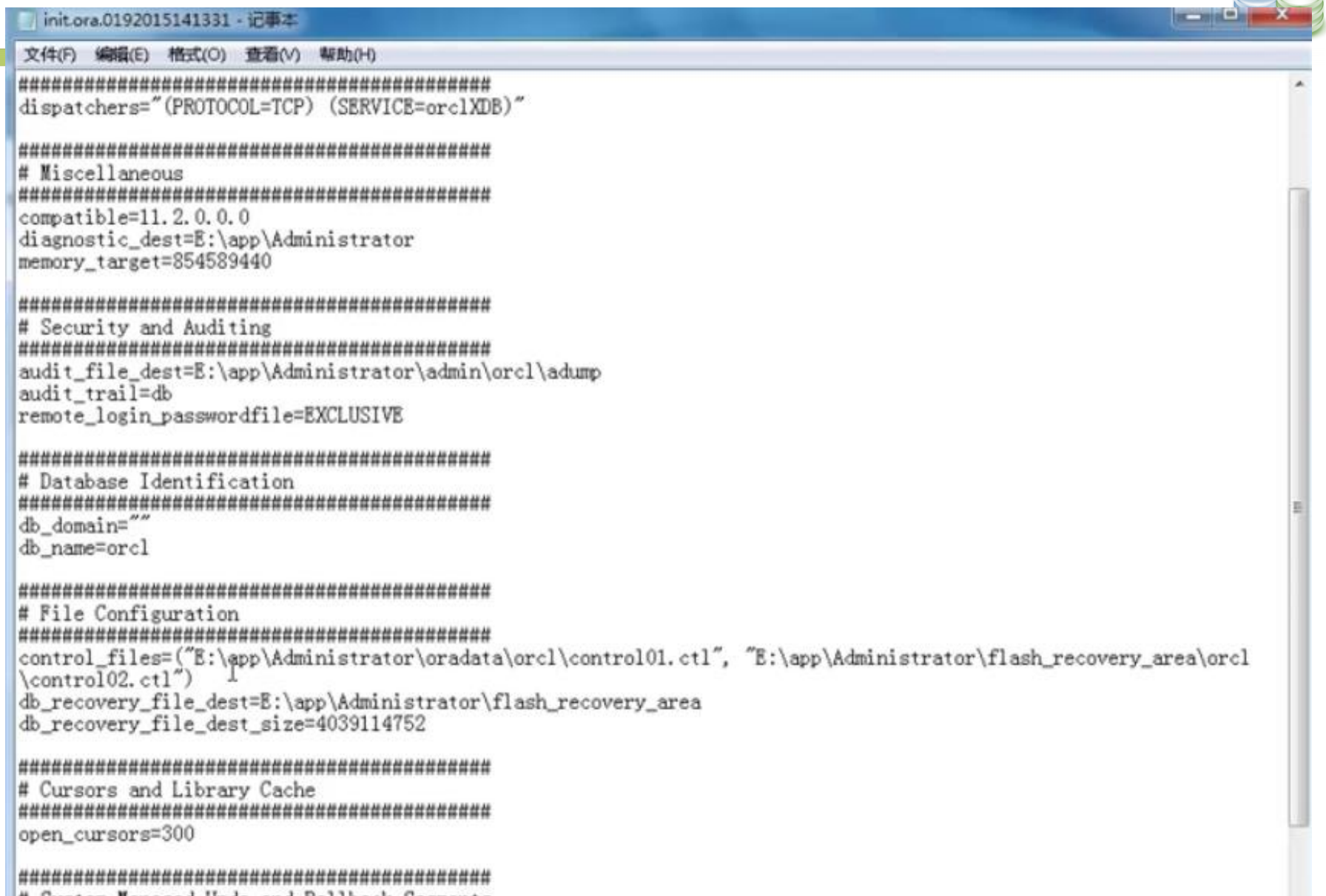
FILE_NAME	TABLESPACE_NAME
E:\APP\ADMINISTRATOR\ORADATA\ORCL\USERS01.DBF	USERS
E:\APP\ADMINISTRATOR\ORADATA\ORCL\UNDOTBS01.DBF	UNDOTBS1
E:\APP\ADMINISTRATOR\ORADATA\ORCL\SYS_AUX01.DBF	SYS_AUX
E:\APP\ADMINISTRATOR\ORADATA\ORCL\SYSTEM01.DBF	SYSTEM

控制文件

数据文件

重做日志文件





```
initora.0192015141331 - 记事本
文件(F) 编辑(E) 格式(O) 查看(V) 帮助(H)
#####
dispatchers="(PROTOCOL=TCP) (SERVICE=orclXDB)"

#####
# Miscellaneous
#####
compatible=11.2.0.0.0
diagnostic_dest=E:\app\Administrator
memory_target=854589440

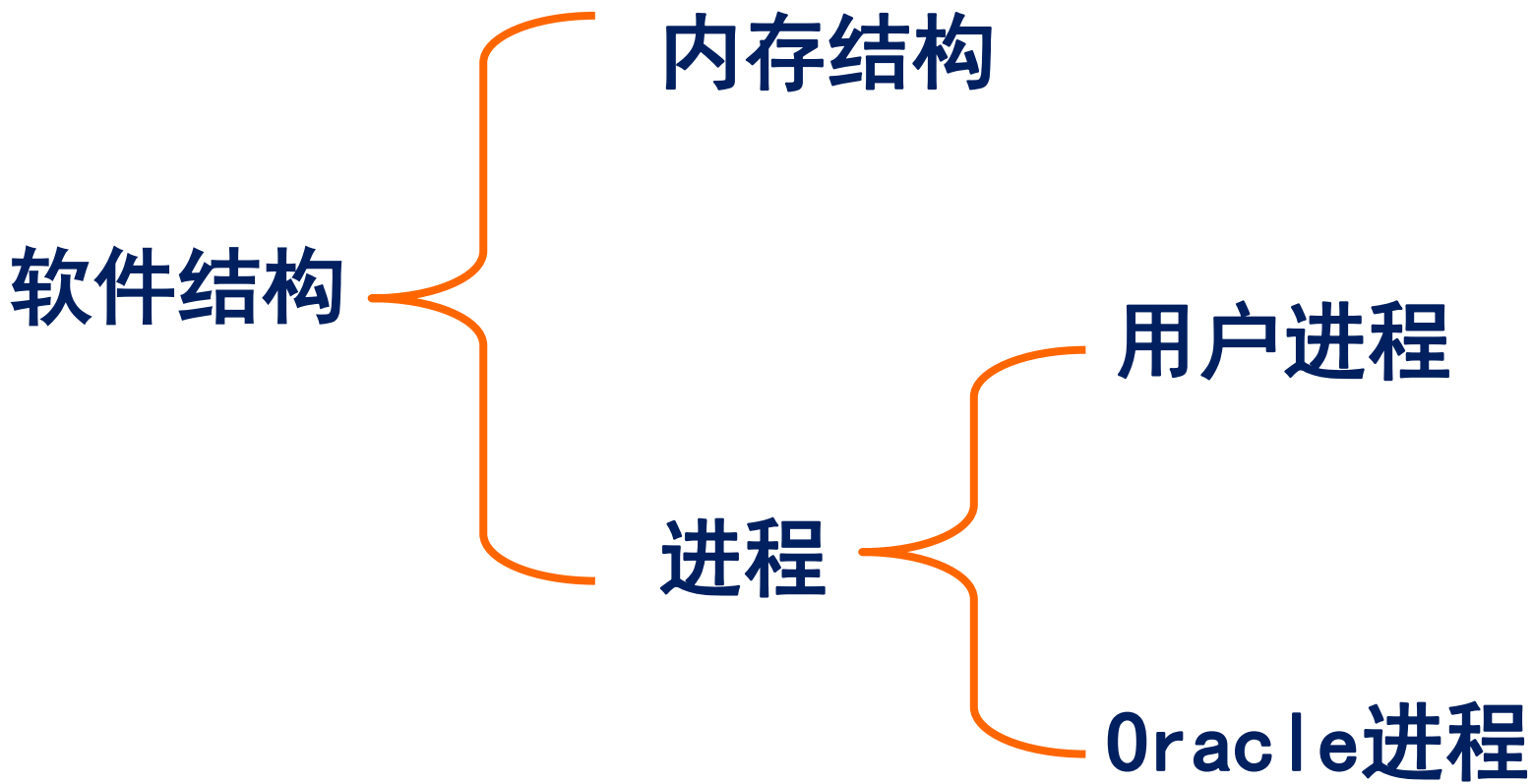
#####
# Security and Auditing
#####
audit_file_dest=E:\app\Administrator\admin\orcl\adump
audit_trail=db
remote_login_passwordfile=EXCLUSIVE

#####
# Database Identification
#####
db_domain=""
db_name=orcl

#####
# File Configuration
#####
control_files=("E:\app\Administrator\oradata\orcl\control01.ctl", "E:\app\Administrator\flash_recovery_area\orcl\control02.ctl")
db_recovery_file_dest=E:\app\Administrator\flash_recovery_area
db_recovery_file_dest_size=4039114752

#####
# Cursors and Library Cache
#####
open_cursors=300

#####
# System Managed Files and Rollback Segments
```

体系结构

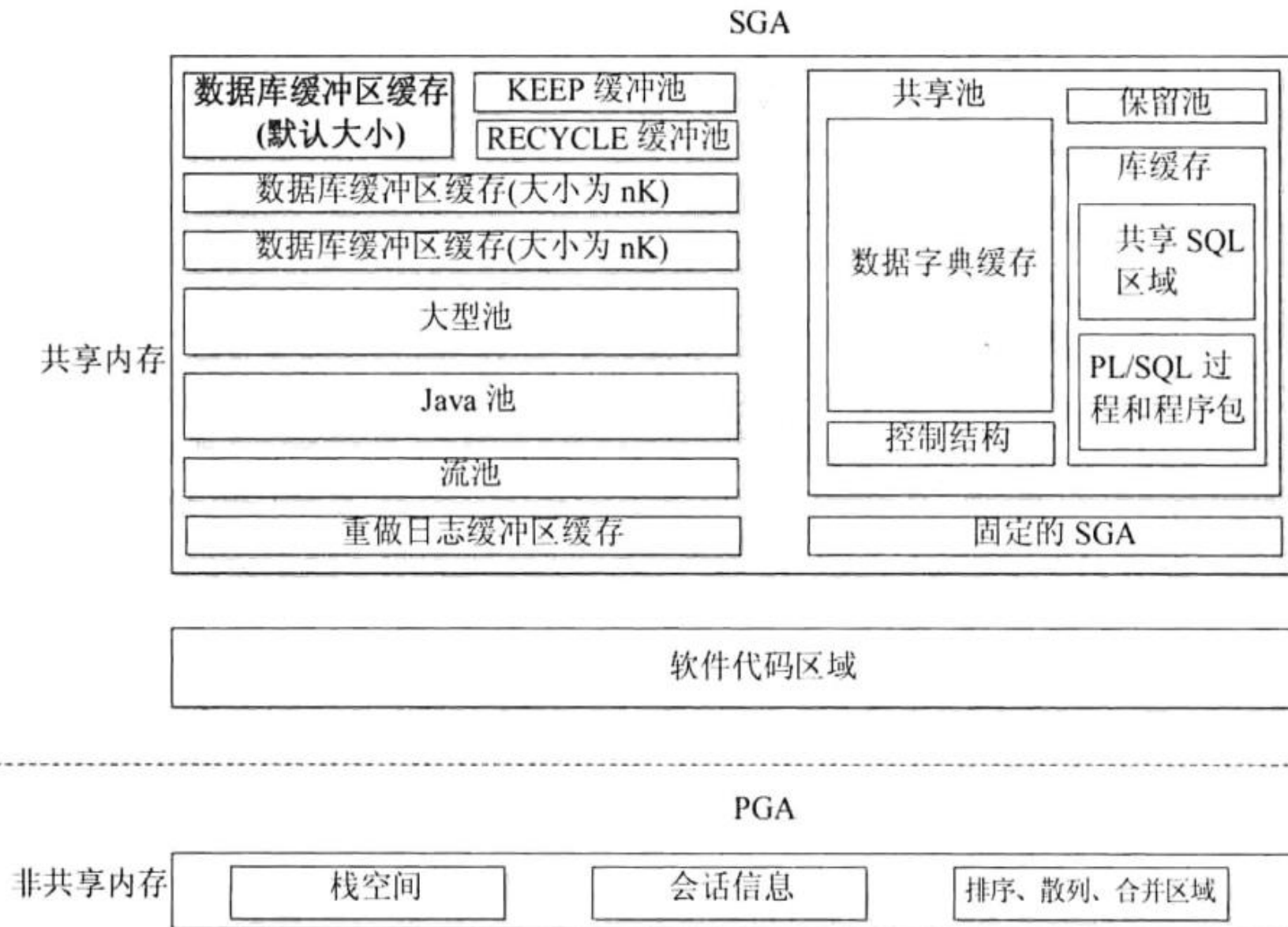
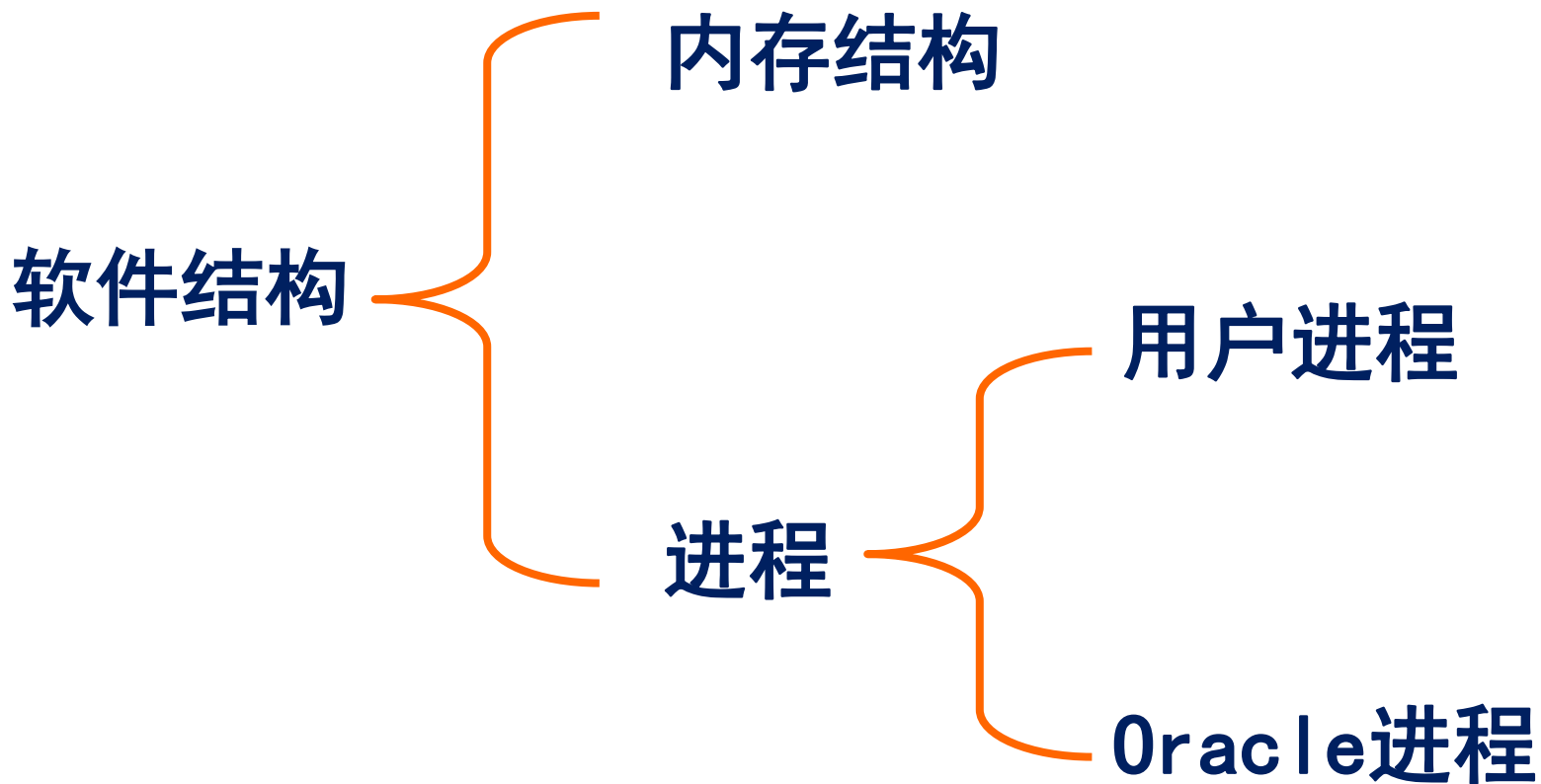
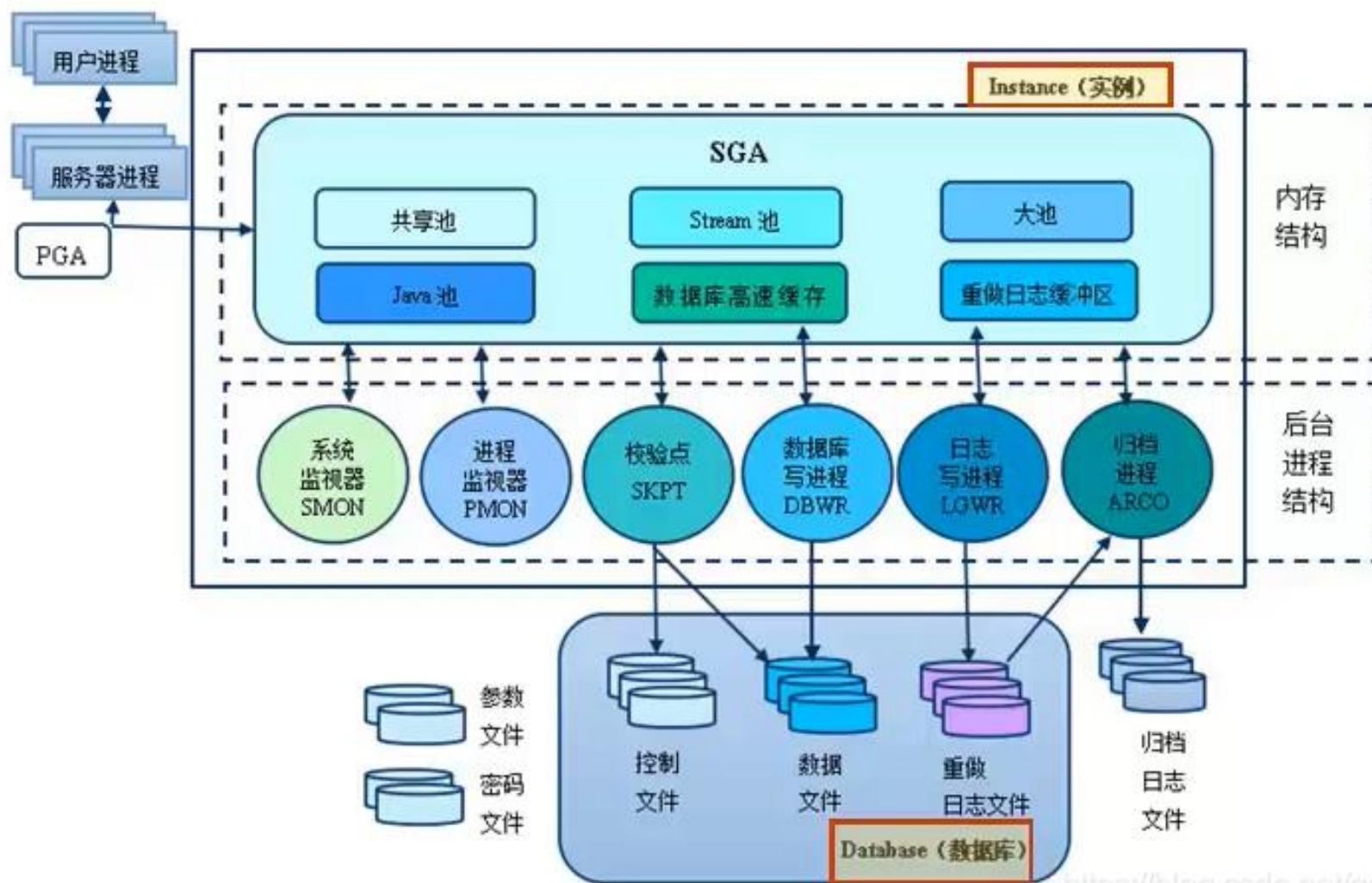


图 1-5 Oracle 逻辑内存结构





<https://blog.csdn.net/gdvfs12>

总体结构



□ 习题

1、下面叙述正确的是（）

- A 一个表空间只能对应一个数据文件
- B 一个数据文件可以对应多个表空间
- C 一个表空间可以对应多个数据文件
- D 数据文件存储了数据库中所有的日志信息



□ 习题

2、用于将数据缓冲区中的数据写到数据文件的进程是（）

A LGWR B DBWn C CKPT D SMON

3、解析后的sql语句会缓存在SGA中的那个区（）

A Java池 B 日志缓冲区 C 数据缓冲区 D 共享池

4、数据库的物理结构不包括下面的那种结构（）

A 日志文件 B 控制文件 C 表空间 D 数据文件



□ 习题

- 5、有关表空间的说法,下列说法不正确的是 ()
- A 从物理上来说, 一个表空间是由具体的一个或多个磁盘上物理文件构成的
 - B 某个用户的数据必定存在于某个表空间中
 - C 表空间是为了统一ORACLE物理和逻辑上的结构而建立的
 - D 表空间的名称可以重复



□ 习题

6 在非归档日志方式下操作的数据库禁用了（）

- A 归档日志
- B 联机日志
- C 日志写入程序
- D 日志文件



- **多租户体系**
- **创建数据库**
- **启动和关闭数据库**
- **创建和修改表**
- **索引和视图**



□ 数据库多租户

在云计算领域中，多租户技术是指多个租户共用一个相同的应用或服务，仍可确保租户之间的数据隔离性。Oracle数据库软件提供了对多租户数据库支持。

一、多租户数据库模式

- 各租户使用各自独立数据库
- 各租户共享数据库，但各自使用隔离Schema
- 各租户共享数据库，共享使用Schema



□ 数据库多租户

相关概念

- **Schema**: 翻译为模式或方案, **Schema** 是数据库对象的集合, 为了区分各个集合, 我们需要给这个集合起个名字, 这些名字可以看作 **schema**, **schema** 里面包含了各种数据库对象, 如 **tables, views, sequences, stored procedures, synonyms, indexes, clusters and database links**。
- Oracle官方文档指出, **schema** 是数据或模式对象的逻辑结构的集合, 由数据库用户拥有, 并且与该用户具有相同的名称, 也就是说 Oracle 的每个用户拥有一个独立的 **schema**。



□ 数据库多租户

三种方案之一

- **独立数据库**：一个租户独享一个数据库实例，这种方案的用户数据隔离级别最高，安全性最好，但成本较高。这种方案与传统的一个客户、一套数据、一套部署类似，差别只在于软件统一部署在运营商。如果面对的是银行、医院等需要非常高数据隔离级别的租户，可以选择这种模式。
 - 优点：为不同的租户提供独立的数据库，有助于简化数据模型的扩展设计，满足不同租户的独特需求；用户**数据隔离级别最高**，安全性最好
 - 缺点：增多了数据库的安装数量，随之带来维护成本和购置成本的增加。



□ 数据库多租户

三种方案之二

- **共享数据库，独立 Schema**：将每个租户关联到**同一个数据库的不同 Schema**，租户间数据彼此逻辑不可见，上层应用程序的实现和独立数据库一样简单。
 - 优点：为安全性要求较高的租户提供了一定程度的逻辑数据隔离，并不是完全隔离；每个数据库可支持更多的租户数量。
 - 缺点：如果出现故障，数据恢复比较困难，因为恢复数据库将牵涉到其他租户的数据；如果需要跨租户统计数据，存在一定困难。



□ 数据库多租户

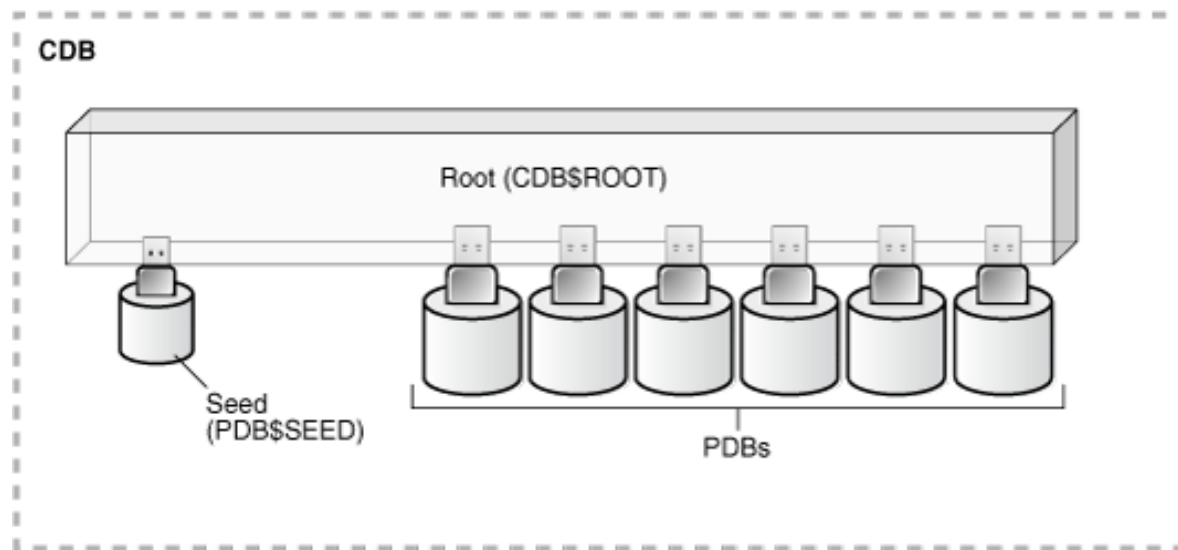
三种方案之三

- **共享数据库，共享 Schema，共享数据表**：租户共享同一个数据库、同一个 Schema，但在表中增加 TenantID 多租户的数据字段。这是共享程度最高、隔离级别最低的模式。每插入一条数据时都需要有一个租户的标识。
 - 优点：维护和购置成本最低，允许每个数据库支持的租户数量最多。
 - 缺点：隔离级别最低，安全性最低，需要在设计开发时加大对安全的开发量；数据备份和恢复最困难，需要逐表逐条备份和还原。



□ Oracle 数据库多租户体系

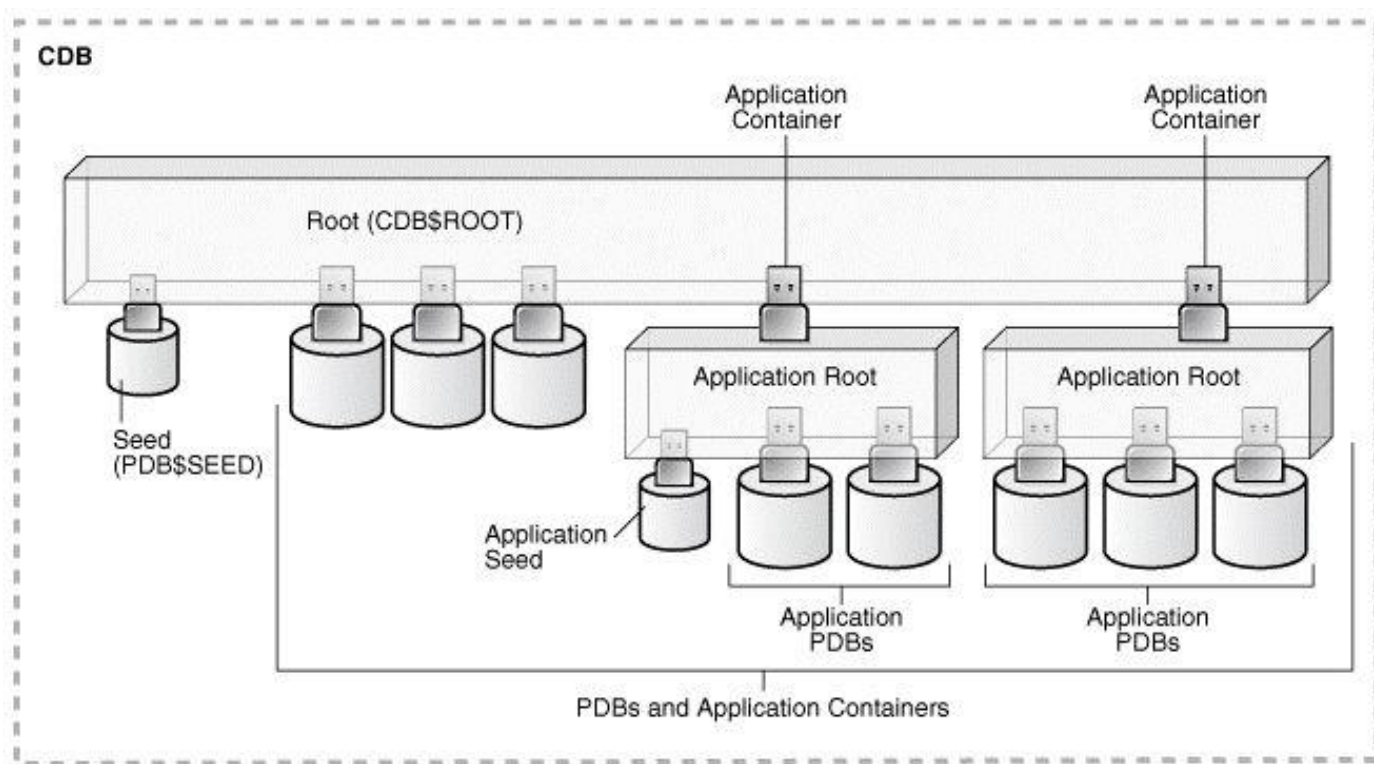
Oracle 数据库可以作为多租户容器数据库 (Multitenant Container Database, CDB) 运行是 Oracle 12c 的新特性，这个特性允许在 CDB 容器数据库中创建并且维护被称为 PDB 的多个数据库，每个 PDB 在 CDB 中是相互独立存在的，在单独使用 PDB 时，与普通数据库无任何区别。12c 之前的所有 Oracle 数据库都是 Non-CDB 数据库。





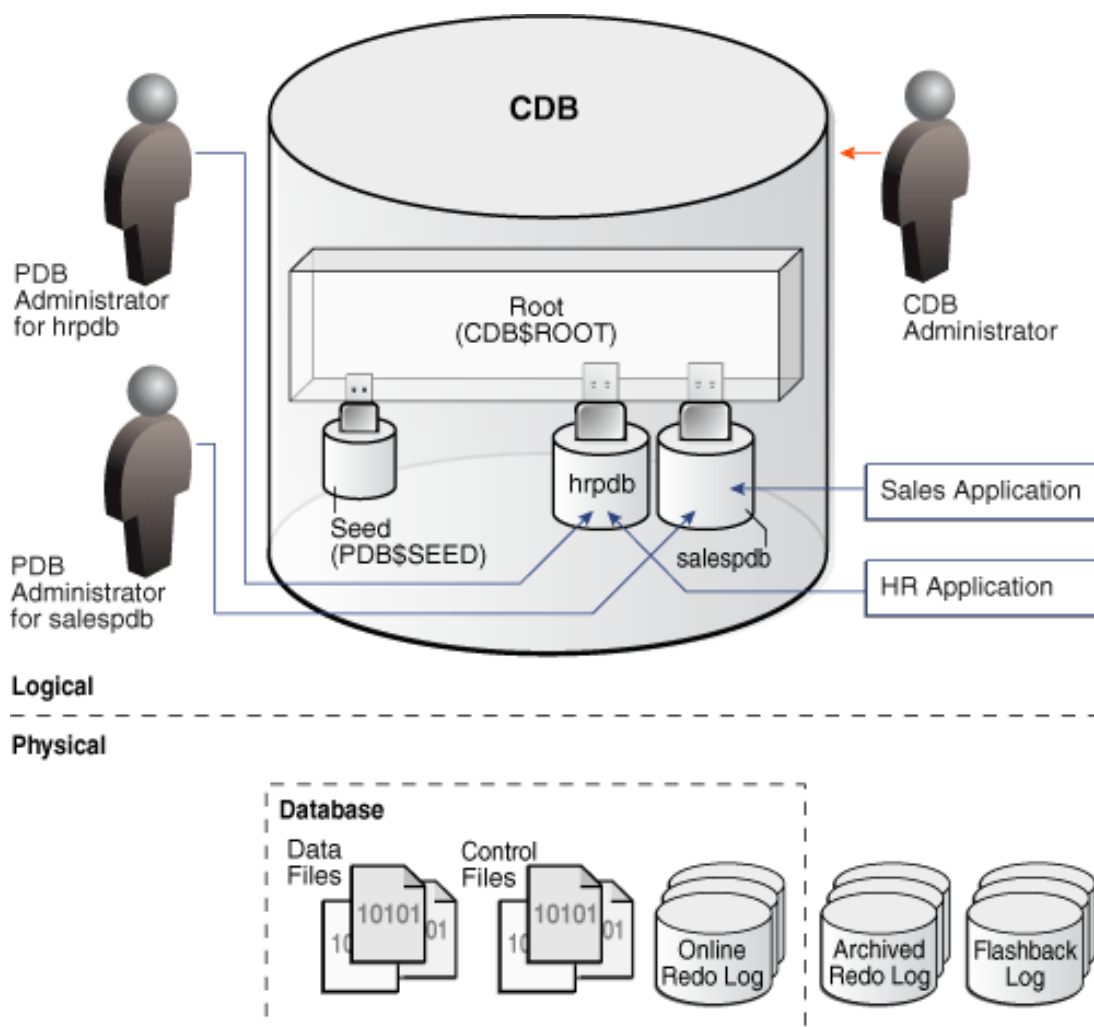
□ Oracle 数据库多租户体系

在 12c R2 版本中，Oracle 对多租户功能进行了增强，在 CDB root 容器中可以创建一个叫做 Application root 的容器，可在其内创建多个依赖于 Application root 的 Application PDBs。





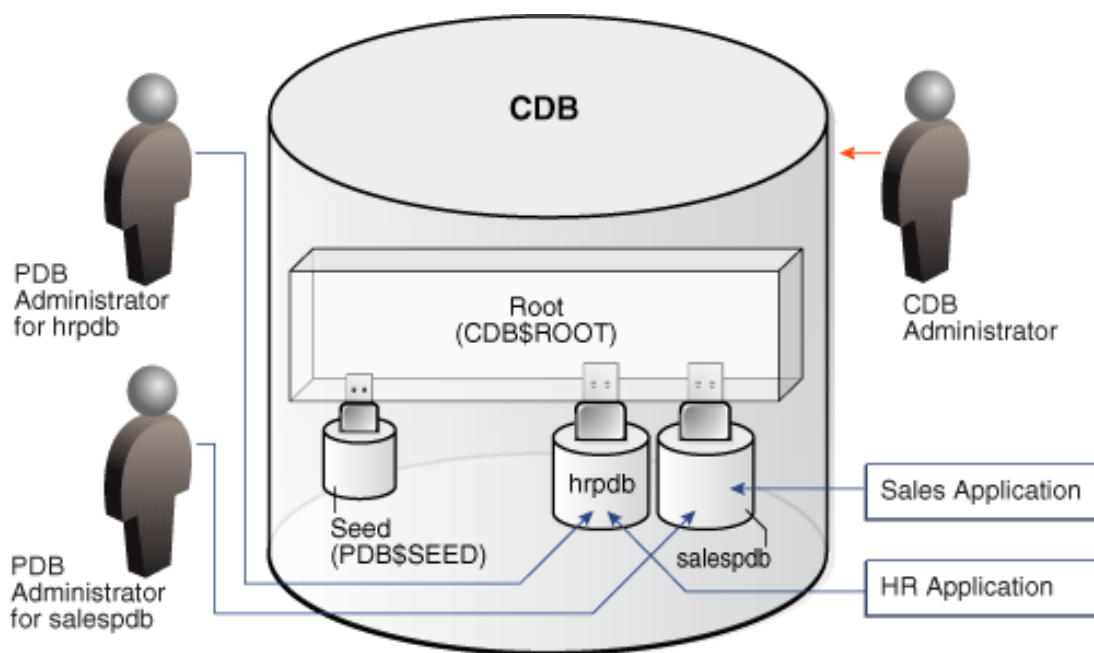
□ Oracle 数据库多租户体系



- **ROOT**: 标识为 **CDB\$ROOT**，是 CDB 环境的根数据库，内含主数据字典视图，其中包含了与 Root 容器有关的元数据和 CDB 中所有的 PDB 信息。每个 CDB 环境中只能有一个 Root 容器数据库。

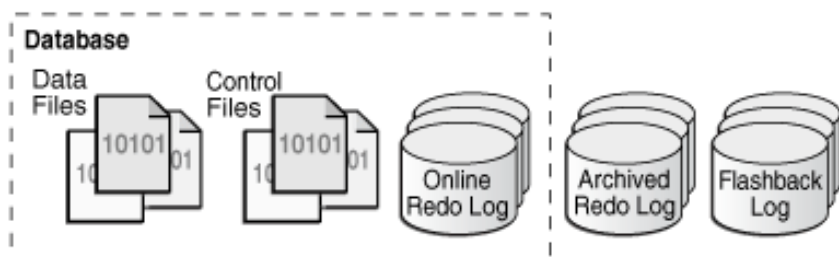


□ Oracle 数据库多租户体系



Logical

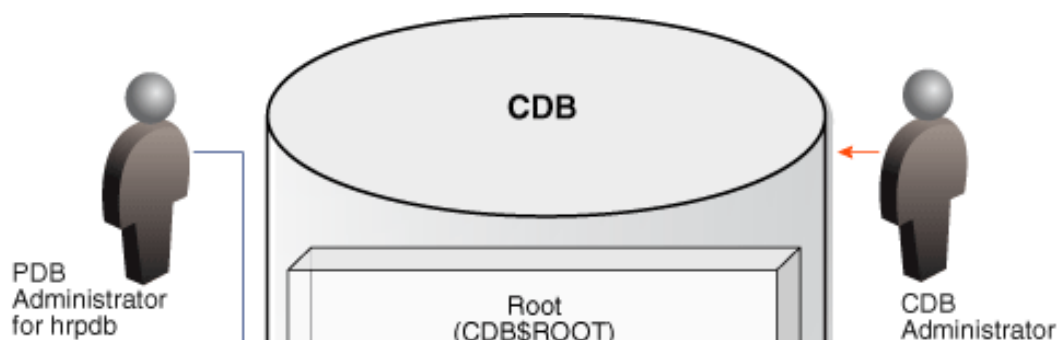
Physical



- **PDB 种子**：标识为 **PDB\$SEED**，是创建 PDB 的模板，可以连接 **PDB\$SEED**，但是不能执行任何操作，因为 **PDB\$SEED** 是只读的，不可进行修改。每个 CDB 环境中只能有一个 PDB 种子容器数据库。



□ Oracle 数据库多租户体系

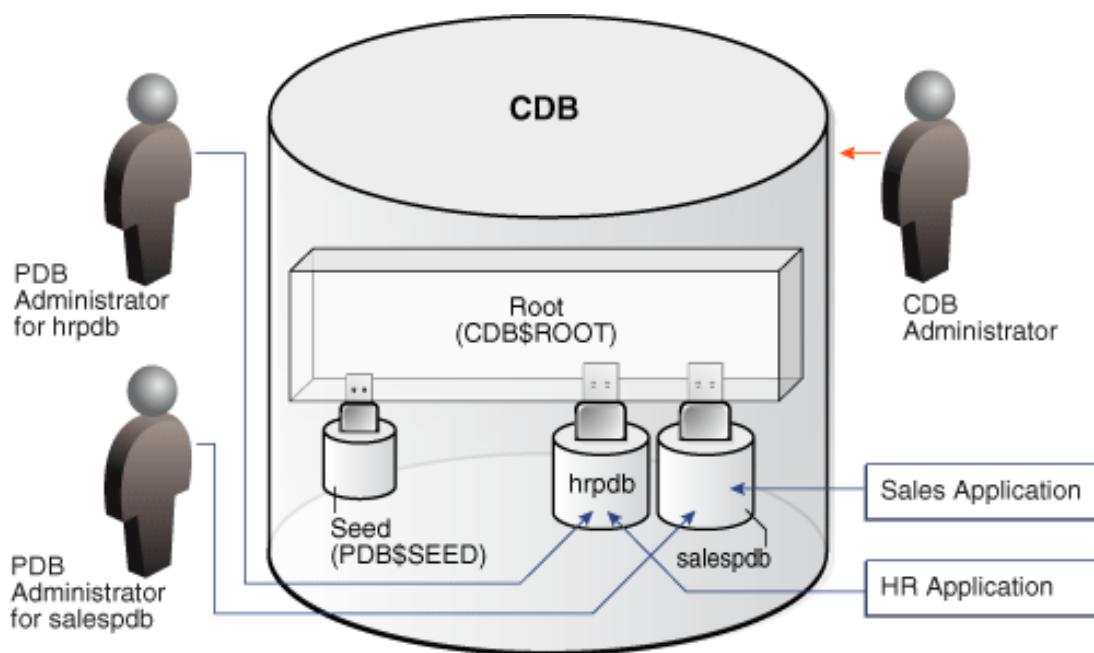


```
sys. ora12c>show pdbs
CON_ID CON_NAME                                OPEN MODE RESTRICTED
-----
2 PDB$SEED                                     READ ONLY NO
3 ORA12CPDB                                    READ WRITE NO
4 PDB2                                         MOUNTED
5 PDB1                                         MOUNTED
6 PDB3                                         READ WRITE NO
sys. ora12c>
```

- **PDB 种子**：标识为 PDB\$SEED，是创建 PDB 的模板，可以连接 PDB\$SEED，但是不能执行任何操作，因为 PDB\$SEED 是只读的，不可进行修改。每个 CDB 环境中只能有一个 PDB 种子容器数据库。

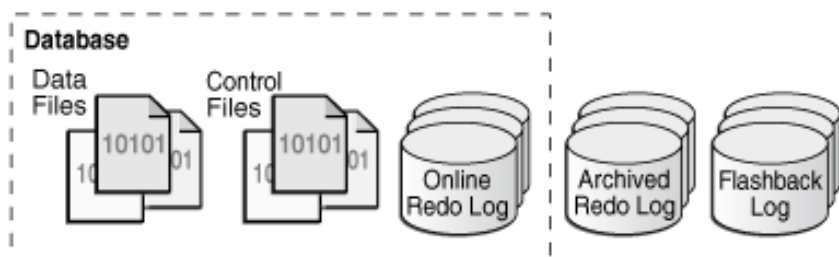


□ Oracle 数据库多租户体系



Logical

Physical



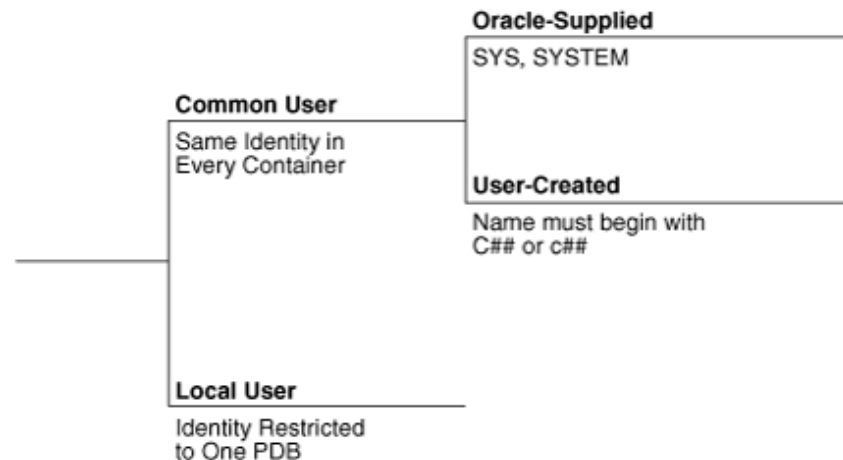
- **PDBs**: 在 CDB 环境中每个 PDB 都是独立存在的，与传统的 Oracle 数据库基本无差别，每个 PDB 拥有自己的数据文件和对象。每个 CDB 环境中有 0 或多个 PDB 容器数据库。



□ Oracle 数据库多租户体系

CDB 环境中的用户

- **Common User**: 在 ROOT 和 PDB 中都有相同的 ID，能连接 ROOT 和所有 PDB 执行操作。Common User 由 Oracle 提供（如 sys、system），也可由用户创建（用户名必须以 C##、c## 开头）。
- **Local User**: 特定于某个 PDB 的，只能在这个 PDB 中操作，不能登录到另一个 PDB 或 ROOT，名称在其 PDB 中是唯一的。
- 具有 SYSDBA 权限的
Common User 可以将
SYSDBA 权限授予
Local User。但用户仍然是本地的。





□ Oracle 数据库多租户体系

CDB 中的授予权限与角色

- **Common User** 授予的权限或角色可以在每个现有的和未来创建的容器中执行。
- **Local User** 授予的权限或角色只能在授予它的容器中执行。
- 用户和角色可以是 **Common** 的，也可以是 **Local** 的。然而，权限本身既不是 **Common** 的，也不是 **Local** 的。如果用户使用 **CONTAINER=CURRENT** 子句在 **Local** 授予权限，那么被授予者具有只能在当前容器中执行的权限。如果用户使用 **CONTAINER=ALL** 子句在 **Common** 授予权限，那么被授予者拥有在任何现有和将来的容器中都可以执行的权限。



□ Oracle 数据库多租户体系

CDB 中的服务

- 创建 PDB 时，数据库将自动在 CDB 中创建 PDB 对应的服务，服务的名称与 PDB 相同。也可以为每个 PDB 创建多个服务。这些服务都将其 PDB 作为初始容器。

```
SQL> SELECT NAME, PDB FROM V$SERVICES ORDER BY PDB, NAME;  
NAME                                PDB  
-----  
SYS$BACKGROUND                      CDB$ROOT  
SYS$USERS                           CDB$ROOT  
db12c                               CDB$ROOT  
db12cXDB                            CDB$ROOT  
db12cbk                             CDB$ROOT  
srvtest1                            CDB$ROOT  
pdb1                                 PDB1  
srvpdb1                             PDB1  
pdb2                                 PDB2  
pdb3                                 PDB3  
srvpdb3                             PDB3
```

PDB1和PDB3存在多个服务



□ Oracle 数据库多租户体系

- **SYSTEM/SYSAUX**: 这两个表空间并不公用，ROOT 以及每个 PDB 都拥有自己的 SYSTEM 和 SYSAUX 表空间。
- **TEMP**: 每个 PDB 都可以有自己的临时表空间，如果没有自己的临时表空间文件，可以使用 ROOT 中的临时表空间。
- **REDO 文件**: 所有 PDB 共用 ROOT 中的 REDO 文件，REDO 中的条目标识 REDO 来自哪个 PDB。
- **归档**: 所有 PDB 共用 CDB 的归档模式和归档文件，不可单独为 PDB 设置自己的归档模式，只有特权用户连接根容器后才能启动归档模式。
- **UNDO MODE**: 在 12.2 之前版本中，所有 PDB 共用 ROOT 中的 UNDO 文件，之后版本 UNDO 的使用模式有两种：SHARED UNDO MODE 和 LOCAL UNDO MODE。



□ Oracle 数据库多租户体系

- **参数文件**：参数文件中只记录了根容器的参数信息，没有记录 PDB 级别的参数信息，在根容器中修改初始化参数，会被继承到所有的 PDB 中，在 PDB 中修改参数后，PDB 的参数会覆盖 CDB 级别的参数，PDB 级别的参数记录在根容器的 `pdb_spfile$` 视图中，但并不是所有的参数都可以在 PDB 中修改，可以通过 `v$system_parameter` 视图查看 PDB 中可修改的参数。
- **控制文件**：所有的 PDB **共用** 一组公共的控制文件，从任何 PDB 中添加数据文件都会记录到公共控制文件当中，Common User 连接根容器时，可对控制文件进行管理。
- **告警日志以及跟踪文件**：所有的 PDB **共用** 一个告警日志和一组跟踪文件，所有的 PDB 告警信息都会写入同一个告警日志中。



□ Oracle 数据库多租户体系

- **时区/字符集**：可以为 CDB 以及所有的 PDB 设置相同的时区/字符集，也可以为每个 PDB 设置单独的时区/字符集。

数据库的字符集

Oracle 字符集

★	US7AASCII
★	ZHS16GBK(ZHT16GBK)

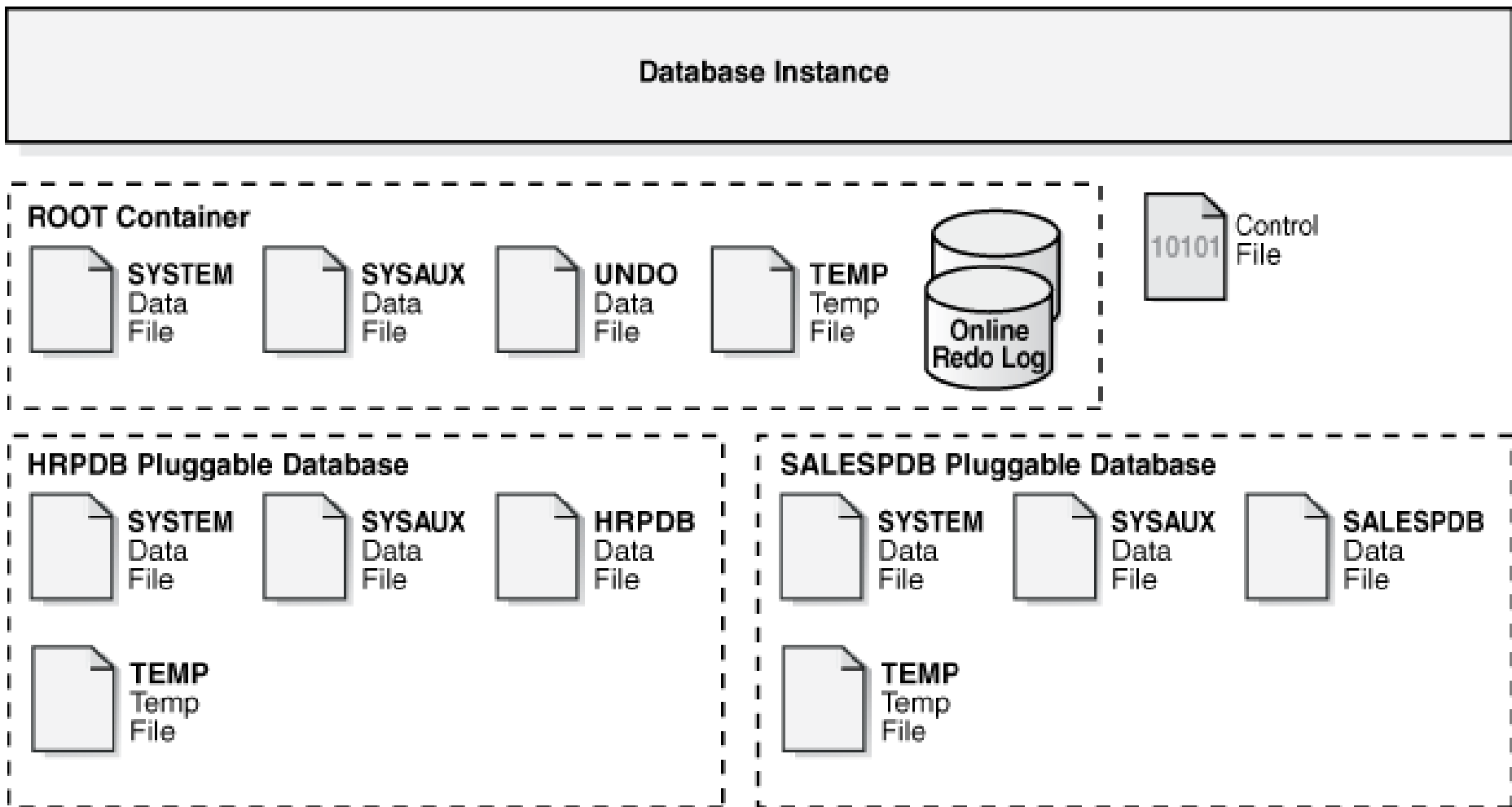
Oracle 国家字符集

★	AL24UTF8SS
★	UTF8
★	AL32FTF8
★	AL16UTF16



□ Oracle 数据库多租户体系

- **数据字典视图与动态性能视图：**在Oracle 12c之前的数据库版本，系统数据字典和用户数据字典采取了混合存放的处理方式。在 CDB 环境中引入了 CDB 级别的数据字典视图，CDB 级别的数据字典视图含有所有 PDB 的元数据信息，其中增加了 con_id 列，con_id 为 CDB 中所有容器唯一标识符，con_id 为 0 的是 ROOT，con_id 为 2 的是 PDB\$SEED，每个 PDB 在 CDB 中都会分配一个唯一的 con_id。如果要想查看 CDB 级别的数据字典视图，必须使用 Common User 在根容器中查看，并且要查看的 PDB 必须处于 open 状态。从用户和应用程序的角度来看，CDB 中每个容器中的数据字典是独立的，就像在非 CDB 中一样。



图中可以看到， CDB 实例可管理多个容器，每个容器（ PDB 、 CDB\$ROOT ）都有自己的 system 、 sysaux 、 temp 等表空间； undo 表空间和 online redo log 通过 CDB\$ROOT 统一管理；每个 PDB 有各自用户表空间；所有容器的表空间对应的数据文件，临时文件分别独立存放和管理；控制文件和参数文件有 CDB 管理；



□ 多租户的优点

- **降低成本**：通过将硬件和数据库基础设施整合到一组后台进程，并有效地共享计算和内存资源，可以降低硬件和维护成本。
- **更容易、更快速地移动数据和代码**：根据设计，可以将 PDB 插入 CDB，将 PDB 从 CDB 中拔出来，然后将这个 PDB 插入另一个 CDB。因此，您可以轻松地将应用程序的数据库后端从一台服务器移动到另一台服务器。
- **更容易管理和监控物理数据库**：CDB 管理员可以通过对用户和 CDB root 执行单个操作（例如打补丁或者执行 RMAN 备份），将环境作为一个集合来管理。



□ 多租户的优点

- 在多租户数据库模式下，各应用的数据库可共享使用一个高性能物理服务器，充分利用该服务器的计算与存储资源。
- 对于机构的IT部门，可集中运维一个数据库服务器，便可将IT应用数据库系统进行统一的数据库备份与恢复等管理工作。

因此，多租户数据库模式不但可以减少机构投入的软硬件成本和数据库系统运行成本，也可以容易实现数据库的灵活扩展、数据迁移，以及安全性隔离。还可以有效地实现数据库服务器集中运行监控和性能调优处理，并减少运维人力资源投入。



- 多租户体系
- 创建数据库
- 启动和关闭数据库
- 创建和修改表
- 索引和视图



□ 方法和类型

创建方法

- 1 使用 DBCA 创建数据库
- 2 使用 CREATE DATABASE 语句创建数据库

创建类型

- 非CDB
- CDB
- PDB

创建数据库



□ 使用 DBCA



创建数据库



□ 使用 DBCA

Database Configuration Assistant - 创建数据库 - 步骤 2/6

创建模式

数据库操作
创建模式
先决条件检查
概要
进度页
完成

☒ 使用默认配置创建数据库(A)

全局数据库名(C):

存储类型(D):

数据库文件位置(E): 浏览...(G)

快速恢复区(I): 浏览...(J)

数据库字符集(K):

管理口令(L):

确认口令(O):

"oracle12c"口令:

☒ 创建为容器数据库(P)

插接式数据库名(Q):

☐ 高级模式(R)

默认配置和高级模式

帮助(H) < 上一步(B) 下一步(N) > 完成(F) 取消

创建数据库



□ 使

Database Configuration Assistant - 管理插接式数据库 - 步骤 4/8

创建插接式数据库

ORACLE 12^c DATABASE

数据库操作
管理插接式数据库
数据库列表
创建插接式数据库
插接式数据库选项
概要
进度页
完成

☒ 创建新的插接式数据库(A)
☐ 从 PDB 档案创建插接式数据库(C)
☐ 使用 PDB 文件集创建插接式数据库(G)

插接式数据库档案(O): 浏览...(E)

插接式数据库元数据文件(I): 浏览...(J)

插接式数据库数据文件备份(K): 浏览...(L)

创建PDB

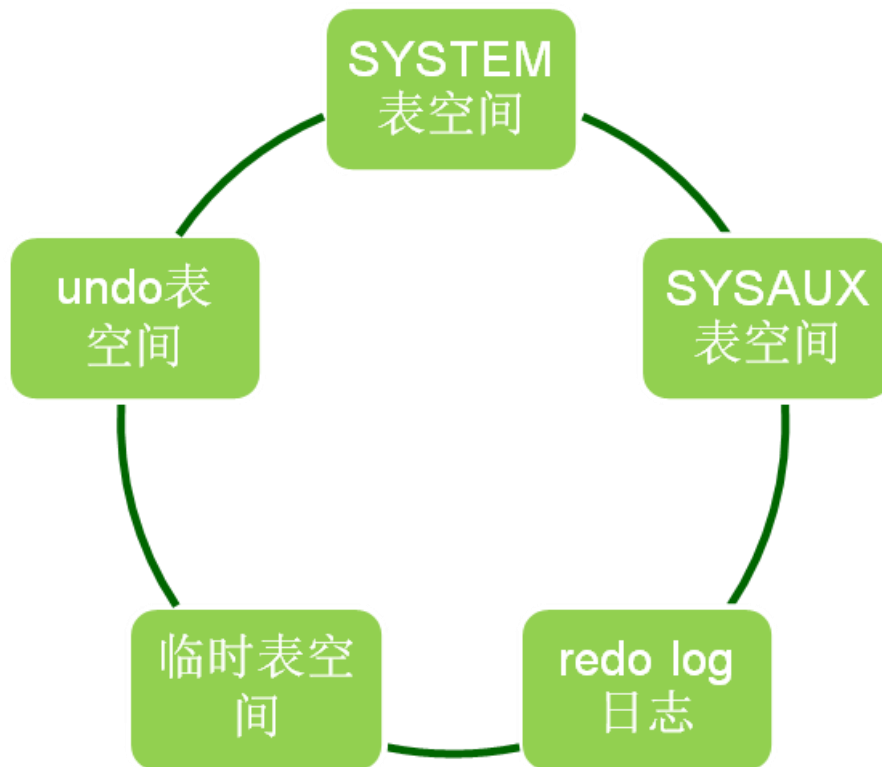
帮助(H) < 上一步(B) 下一步(N) > 完成(F) 取消



□ 使用 CREATE DATABASE

以创建非 CDB (Non-CDB) 数据库为例, 说明创建步骤。

一个oracle数据库, 最基本的构成



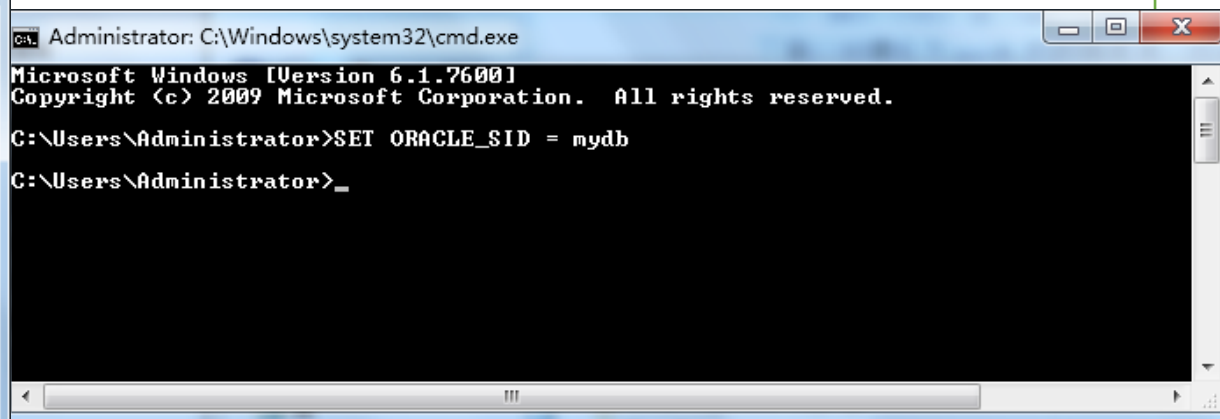
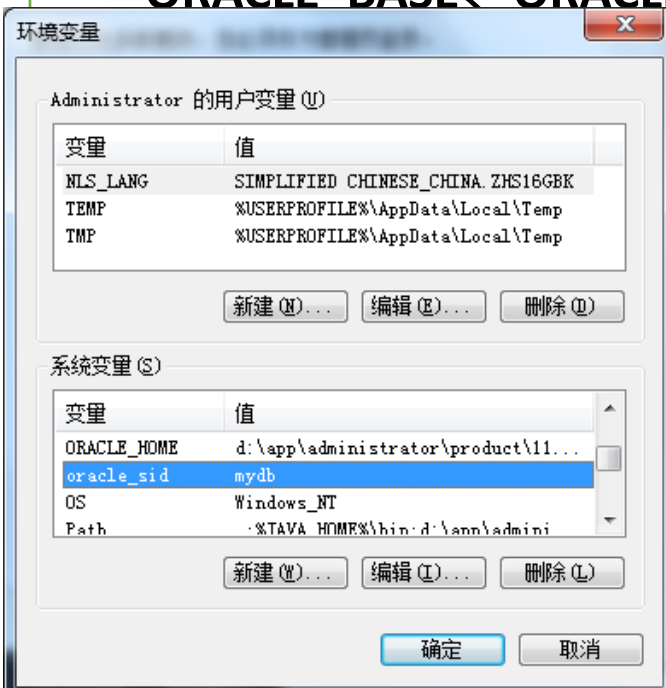
创建数据库



□ 使用 CREATE DATABASE

以创建非 CDB（Non-CDB）数据库为例，说明创建步骤。

1. **指定实例标识符（SID）**：用于该实例从稍后可能创建并同时在同一主机上运行的其它 Oracle 数据库实例区分开来；
2. **设置所需的环境变量**：确保 ORACLE_UNQNAME、ORACLE_SID、ORACLE_BASE、ORACLE_HOSTNAME、ORACLE_HOME 等环境变量设置





□ 使用 CREATE DATABASE

以创建非 CDB (Non-CDB) 数据库为例，说明创建步骤。

1. **指定实例标识符 (SID)**：用于该实例从稍后可能创建并同时在同一主机上运行的其它 Oracle 数据库实例区分开来；
2. **设置所需的环境变量**：确保 ORACLE_UNQNAME、ORACLE_SID、ORACLE_BASE、ORACLE_HOSTNAME、ORACLE_HOME 等环境变量设置正确；
3. **选择数据库管理员身份验证方法**；
4. **创建初始化参数文件**：当 Oracle 实例启动时，它读取一个初始化参数文件。可仿照 Orcl 实例的初始化参数文件创建；
5. **(仅限 Windows) 创建实例**：在 Windows 平台上，连接到实例之前，必须手动创建它；



□ 使用 CREATE DATABASE

6. **连接到实例**：启动 SQL * Plus 并使用管理权限连接到 Oracle 数据库实例 SYSDBA；
7. **创建服务器参数文件**：服务器参数文件使用户可以使用该 ALTER SYSTEM 命令更改初始化参数，并在数据库关闭和启动时保留更改；
8. **启动实例**：启动实例而不连接数据库（startup nomount）；
9. **运行 CREATE DATABASE 语句**；
10. **创建其他表空间**：要使数据库起作用，须为应用程序创建其他表空间。
11. **运行脚本以构建数据字典视图**：运行构建数据字典视图，同义词和 PL / SQL 包所必需的脚本，并支持 SQL * Plus 的正常运行；
12. **(可选)** 运行脚本以安装其他选项、备份数据库、启用自动实例启动。



□ 使用 CREATE DATABASE

创建 CDB 和 Non-CDB 的差异:

- CDB 数据库创建时应指定 **ENABLE PLUGGABLE DATABASE** 子句;
- CDB 数据库应指定根文件和种子文件的名称和位置;
 - 使用 **seed FILE_NAME_CONVERT** 子句
 - 基于 Oracle 文件托管方式 (OMF)
 - 使用 **PDB_FILE_NAME_CONVERT** 初始化参数
- 可以为 PDB 种子数据库制定不同的属性;
 - 根 **SYSTEM** 和 **SYSAUX** 表空间的数据文件的属性可能不适合种子。
可以为种子的数据文件指定不同的属性 **tablespace_datafile**。

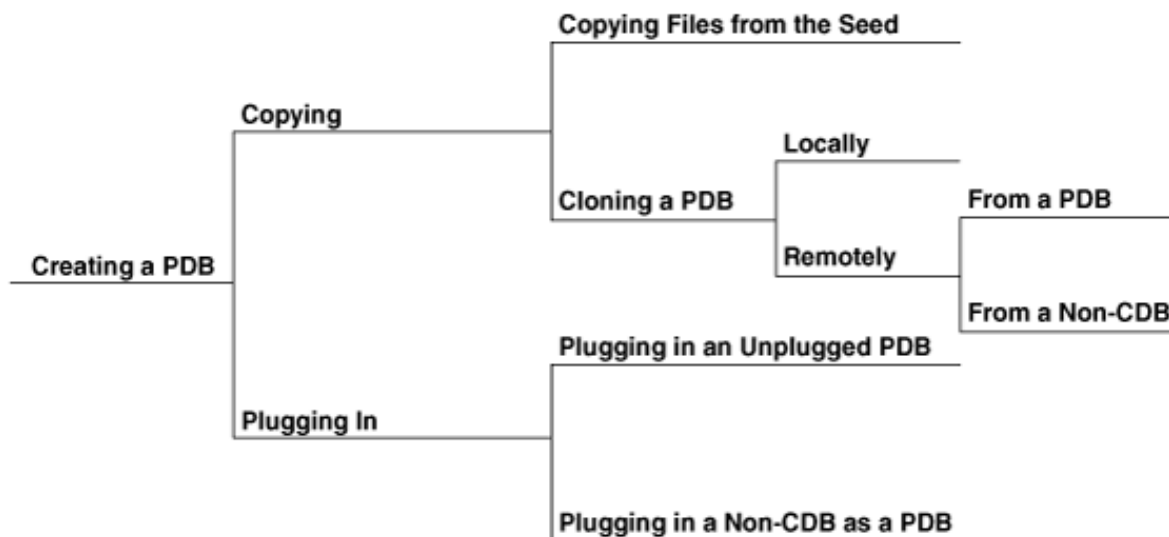


□ 使用 CREATE DATABASE

创建 PDB 的前提：

- CDB 必须存在，且处于读 / 写模式。
- 当前用户必须是 ROOT 容器的 Common User 。
- 当前用户必须具有 CREATE PLUGGABLE DATABASE 系统权限。
- 必须为每个 PDB 确定唯一的 PDB 名称。

创建 PDB 的几种方法：





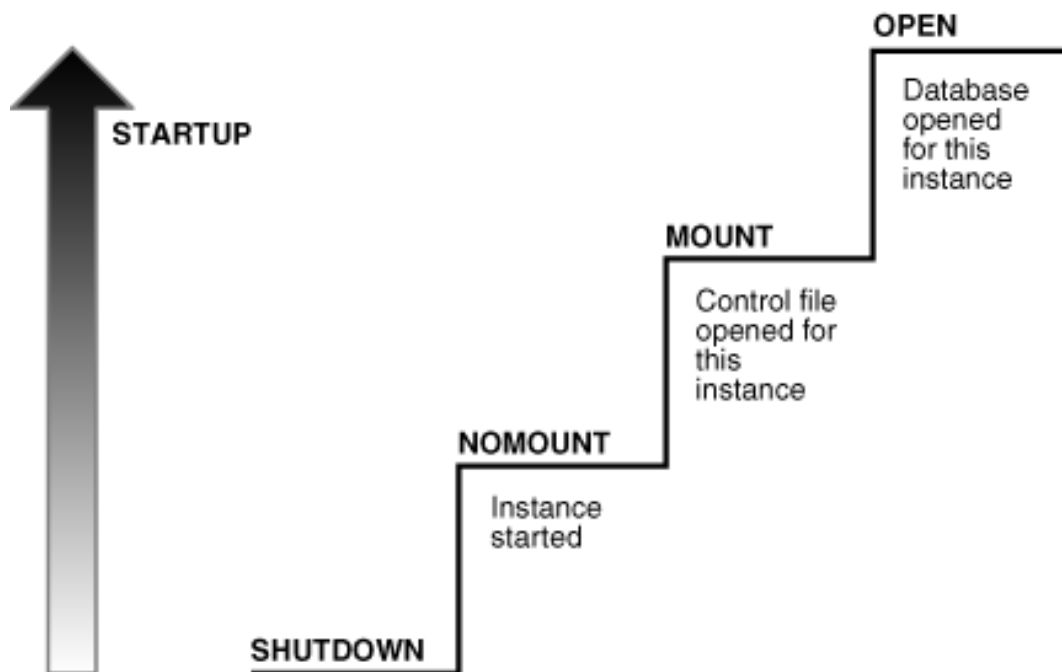
- 多租户体系
- 创建数据库
- 启动和关闭数据库
- 创建和修改表
- 索引和视图

启动和关闭数据库



□ 启动数据库

启动步骤



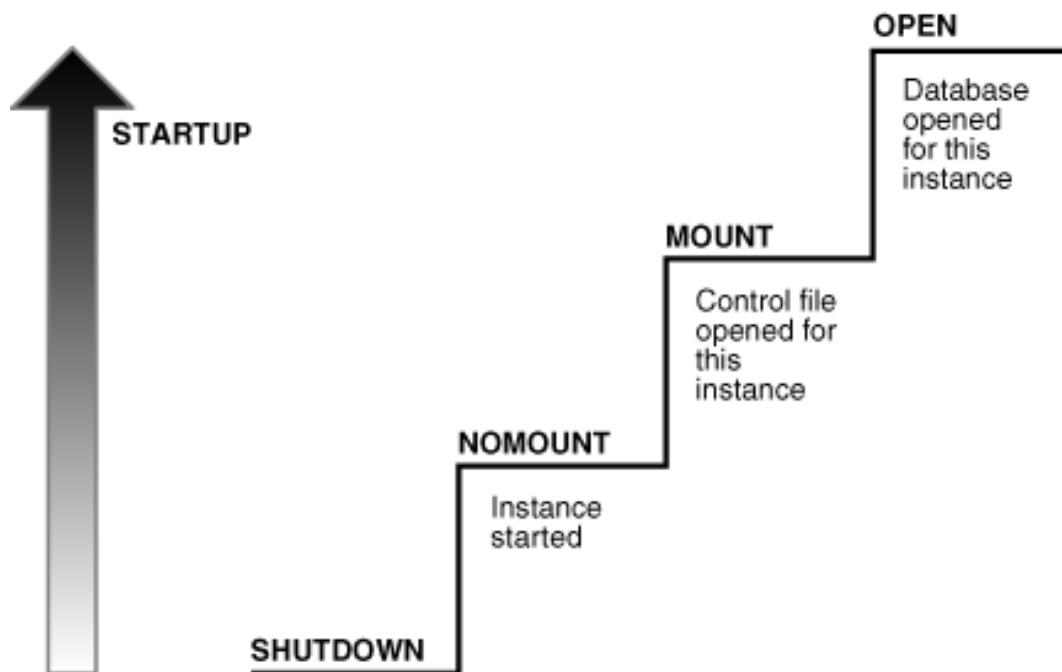
NOMOUNT 状态：启动实例，读取初始化文件，分配 SGA，启动后台进程，启动预警日志文件，但是没有与数据文件关联。

启动和关闭数据库



□ 启动数据库

启动步骤



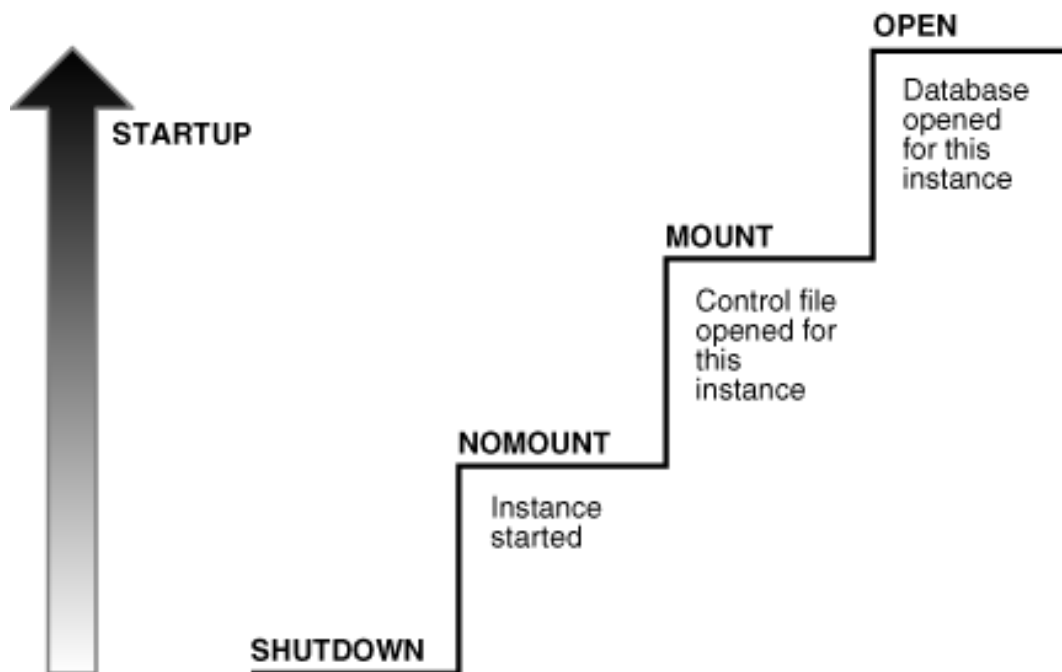
MOUNT 状态： 关联实例与数据库，读取控制文件并获取数据文件和重做日志文件名称状态，但是用户不可用。

启动和关闭数据库



□ 启动数据库

启动步骤

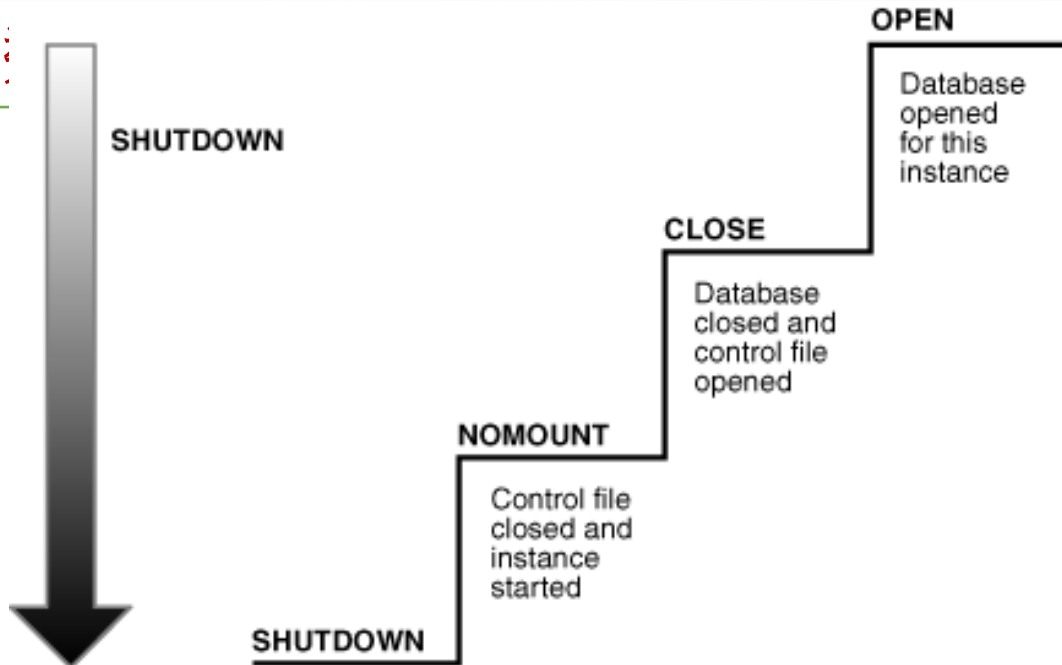


OPEN 状态：打开数据文件，打开重做日志文件，检验数据库的一致性，若不一致，SMON 后台进程将启动实例恢复。授权用户可以访问数据库。

启动和关闭数据库



□ 关闭



阶段	加载状态	描述
1	关闭数据库	数据库处于加载状态，但是在线数据文件和重做日志文件被关闭。
2	卸载数据库	实例处于启动状态，但是不再关联数据库的控制文件。
3	关闭实例	实例被关闭，不再处于启动状态。



- 多租户体系
- 创建数据库
- 启动和关闭数据库
- 创建和修改表
- 索引和视图



□ 表结构设计

- 确定表与列的命名
 - 长度、字符使用、唯一性符合要求；
- 确定列的类型
 - 数字、字符、大对象、日期和时间、二进制、行；
- 确定列的约束
 - NOT NULL、UNIQUE、CHECK、PRIMARY KEY、FOREIGN KEY；

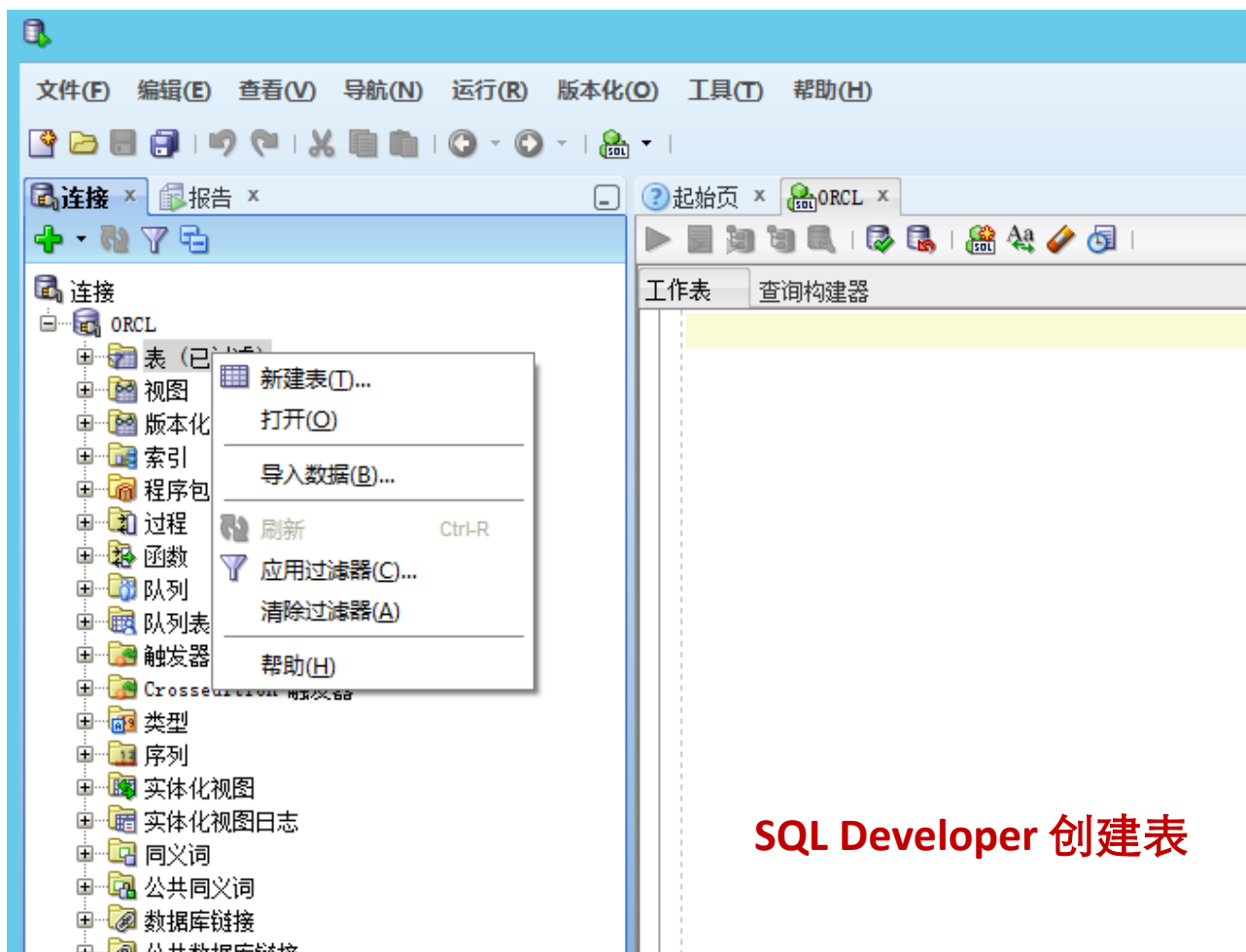
□ 创建表和修改表结构的 SQL 语句

- CREATE TABLE
- ALTER TABLE

创建和修改表



□ 创建表



SQL Developer 创建表

创建和修改表



□ 创建表

创建表

方案(S): SYSTEM

名称(M): TABLE1

表类型: ☒ 正常(N) ☐ 外部(X) ☐ 按索引组织(I) ☐ 临时(事务处理)(P) ☐ 临时(会话)(T)

列(C):

- COLUMN1

列属性:

名称(A): COLUMN1

数据类型: ☒ 简单(E) ☐ 复杂(L)

类型(Y): VARCHAR2

大小(Z): 20

单位(U):

默认(D):

☐ 不能为 NULL(E)

注释(O):

帮助(H) 确定 取消

SQL Developer 创建表时列的高级设置



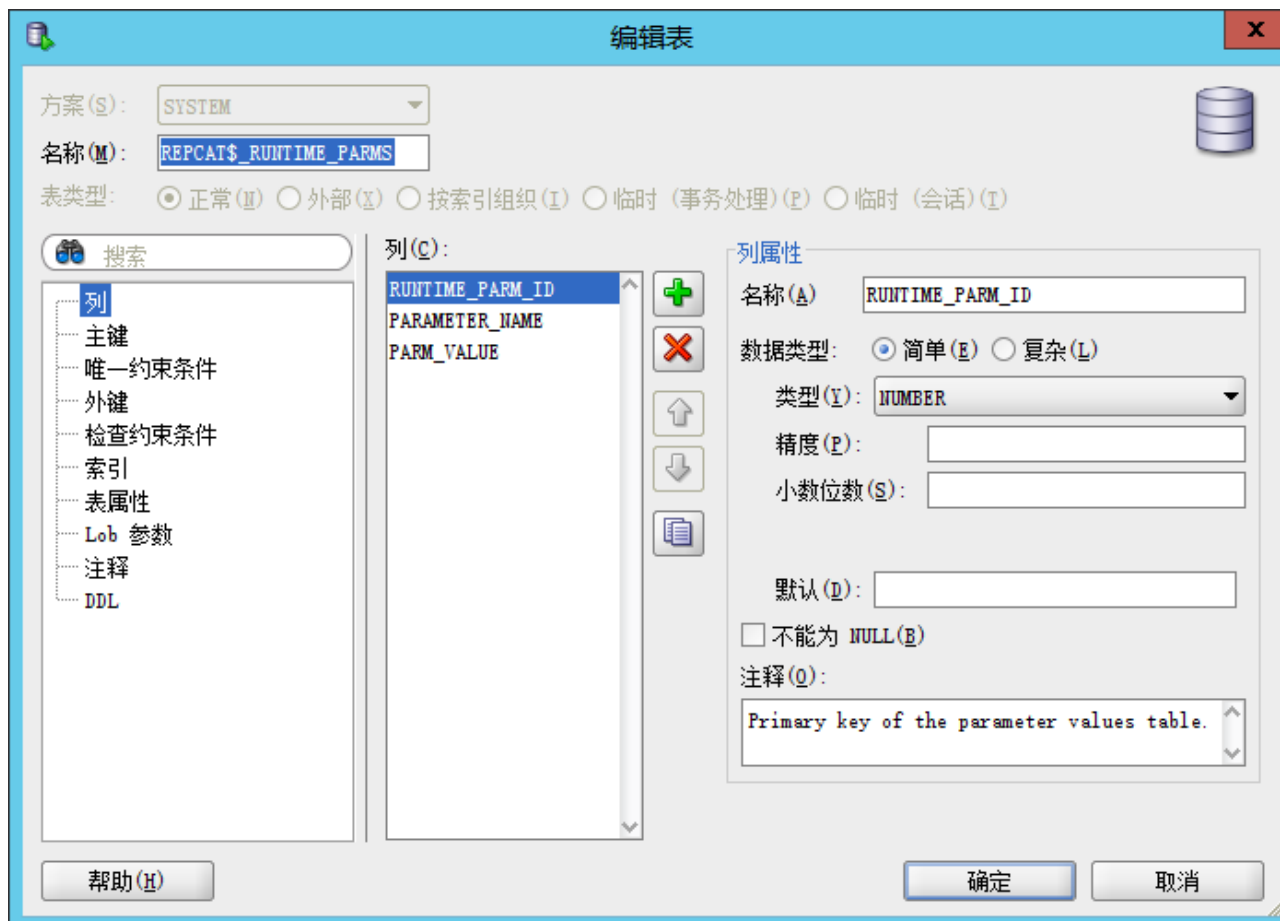
□ 修改表结构

- 增加列
- 更改列
 - 数据类型、长度、默认值、约束；
- 直接删除列
- 将列标记为 **UNUSED** 状态

创建和修改表



□ 修改表结构



SQL Developer 更改表列



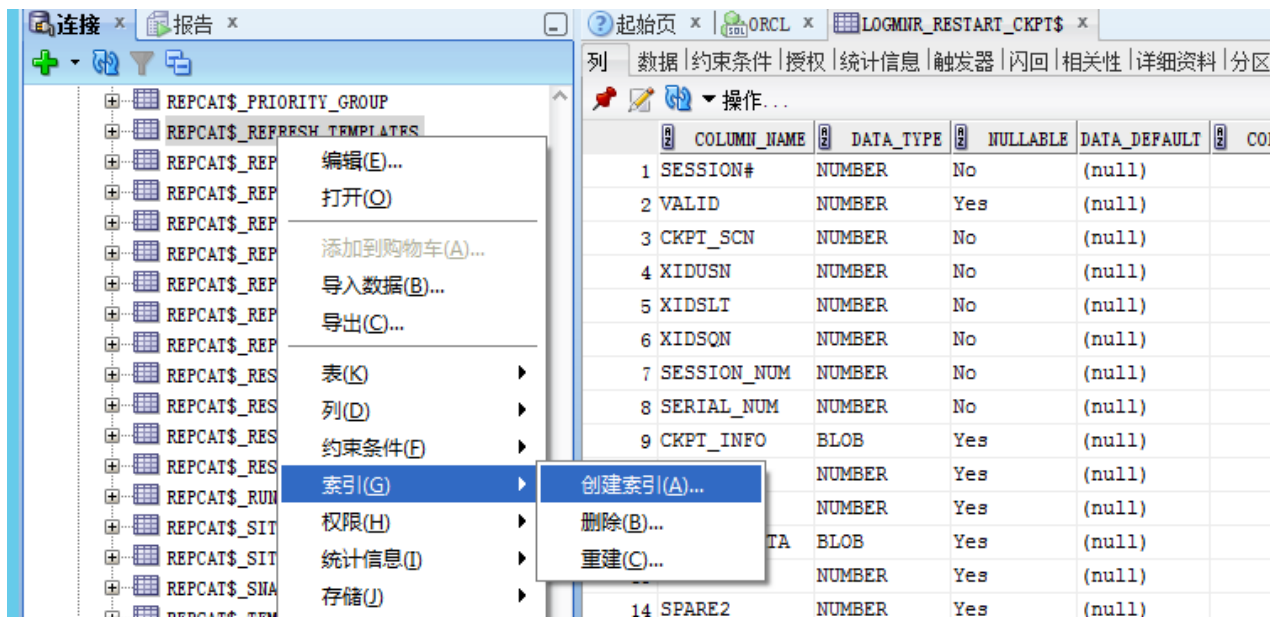
- 多租户体系
- 创建数据库
- 启动和关闭数据库
- 创建和修改表
- 索引和视图



□ 创建索引

索引是数据库对象之一，用于提高查询效率。索引的内建工作对用户是透明的，由数据库自行维护，只需要指定是否添加索引。索引是为表中字段添加的。当一个字段经常出现在 WHERE 中作为过滤条件，或 ORDER BY 或 DISTINCT 中时可以为它添加索引以提高查询效率。

SQL 语句：CREATE INDEX



SQL Developer

创建索引



□ 创建索引



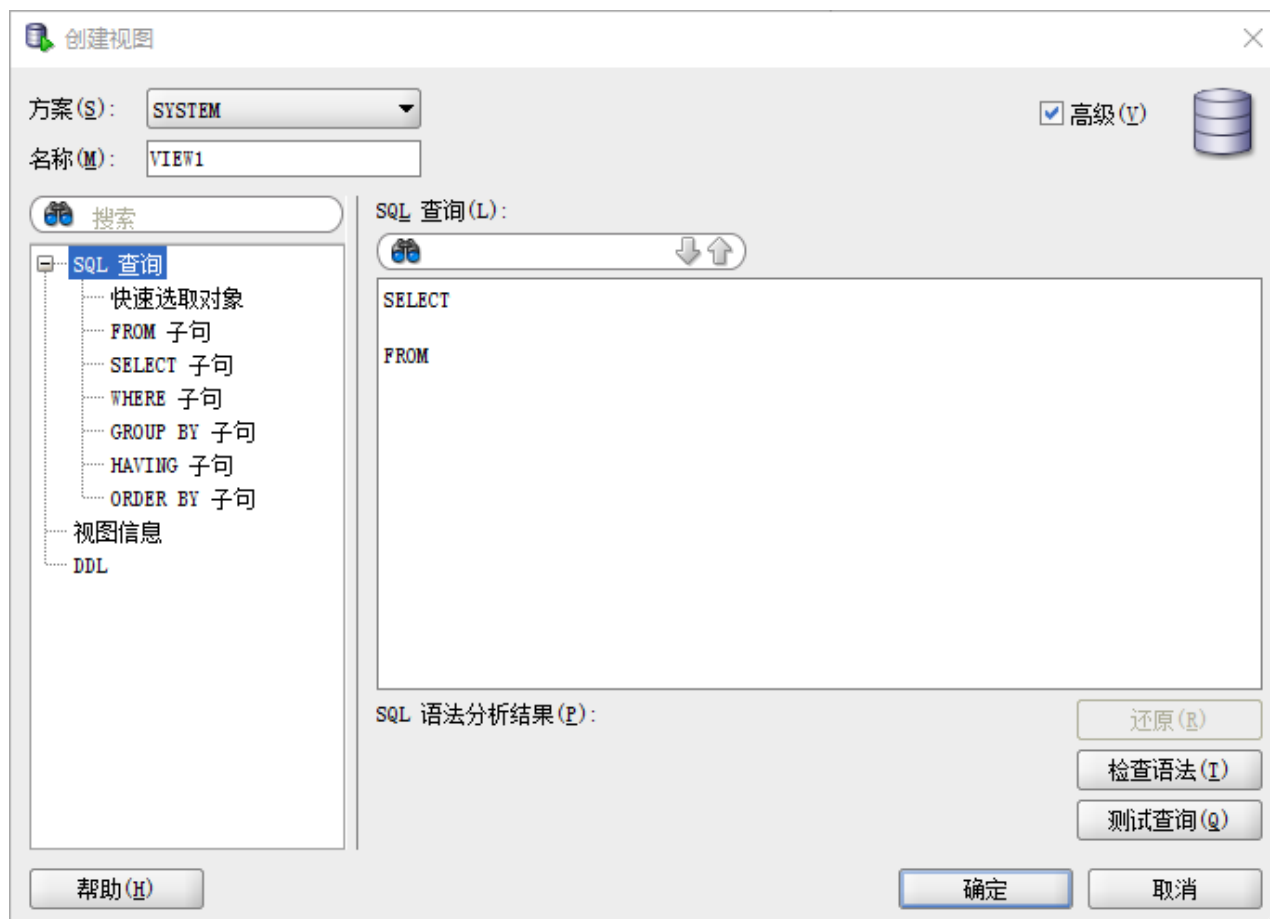
□ 创建视图

视图也是数据库对象之一，它是一个虚拟表，其内容由查询定义。同真实的表一样，视图包含一系列带有名称的列和行数据。但是，视图并不在数据库中以存储的数据值集形式存在。行和列数据来自定义视图的查询所引用的表，并且在引用视图时动态生成。





□ 创建视图



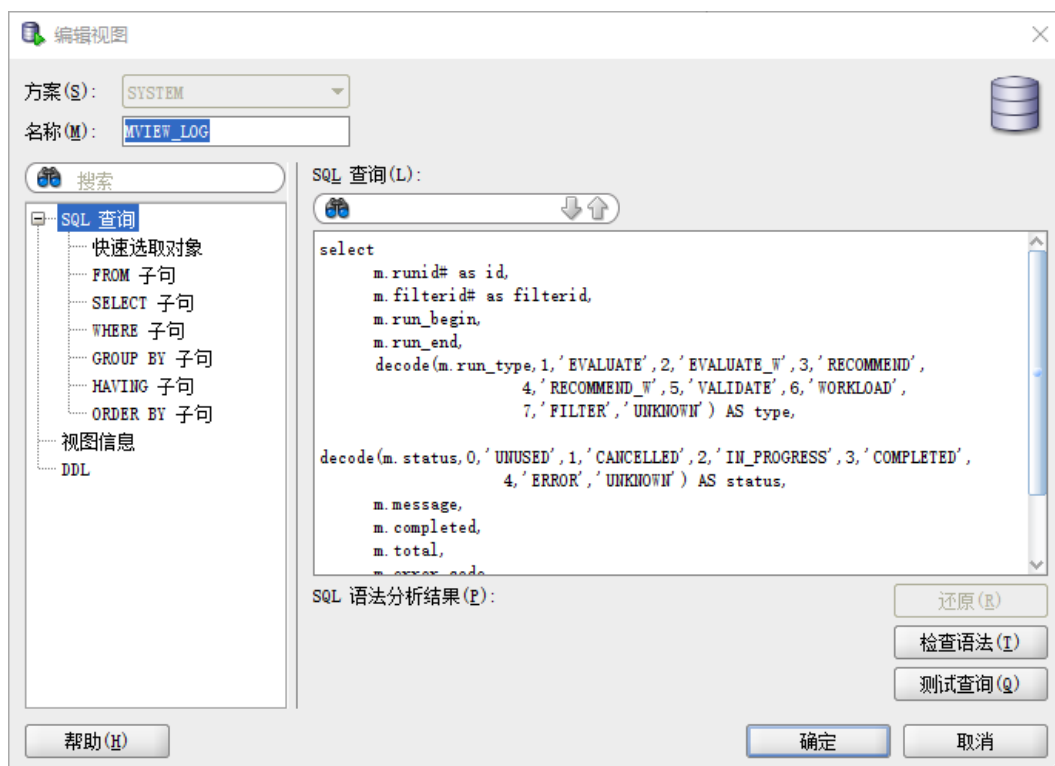
SQL Developer 创建视图的高级选项



□ 更改视图

- 视图只是一个虚表，没有数据，更改视图只是改变数据字典对该视图的定义信息，视图中的所有基础对象的定义和数据都不会收到任何影响。
- 更改视图之后，依赖于该视图的所有视图和 PL/SQL 程序都将变为失效状态。

SQL Developer 更改视图





1. 掌握 Oracle 多租户架构，了解 CDB 环境的逻辑结构和物理结构。
2. 熟悉 DBCA 使用，掌握使用 DBCA 创建非 CDB、CDB 和 PDB 的方法。
3. 掌握使用 SQL Developer 对表、索引、视图的创建和修改方法。
4. 自学使用 SQL Developer 实现表空间的创建、修改、删除



□ 习题

创建数据库时，Oracle如何得知需要创建的控制文件信息（）

- A 从初始化参数文件
- B 从 CREATE DATABASE 命令
- C 从环境变量
- D 从\$ORACLE_HOME目录名为 <db_name>.ctl的文件