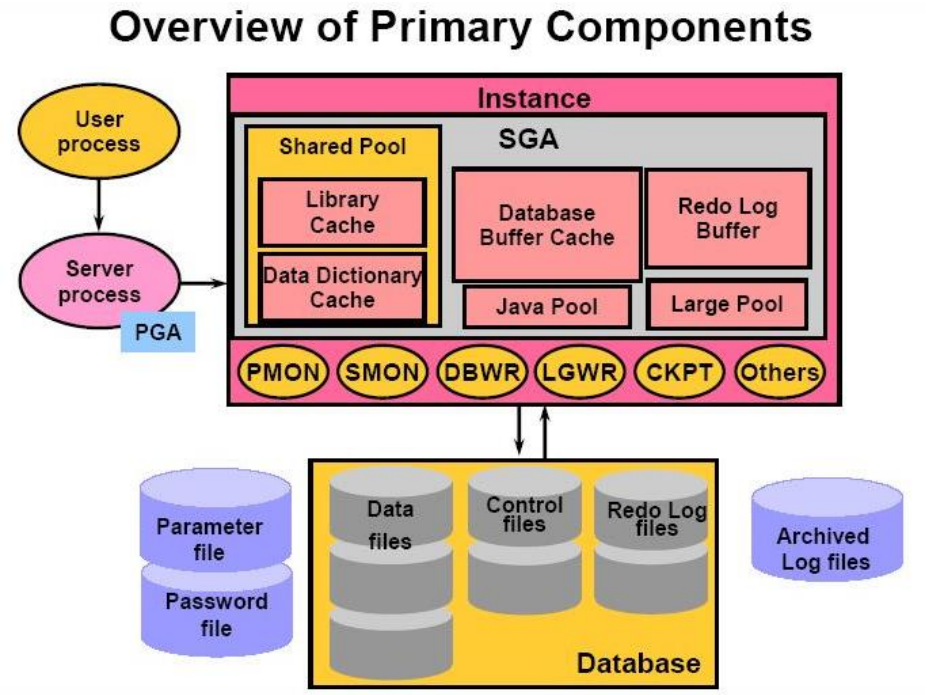
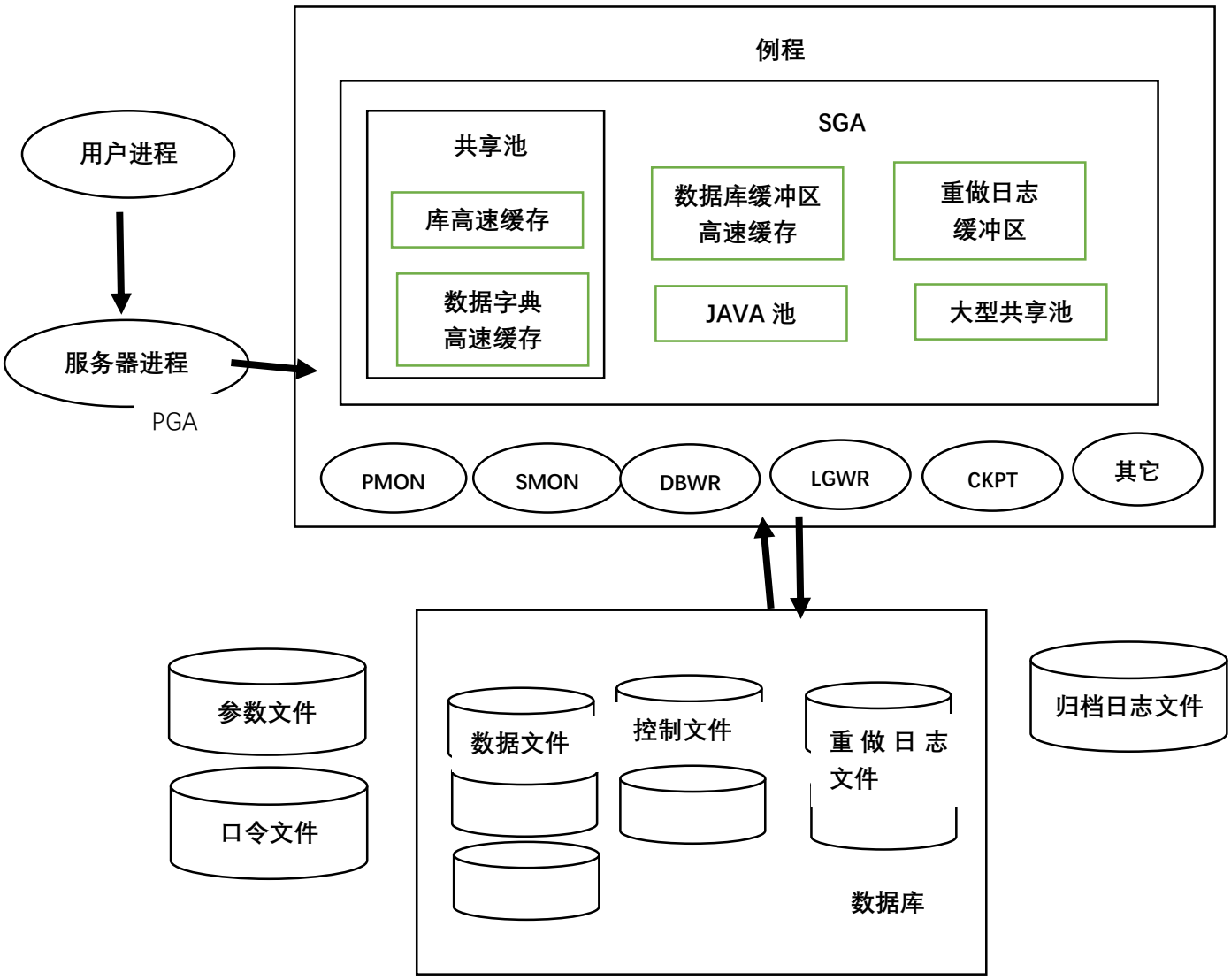
1. Oracle 体系结构





Oracle 体系结构包括很多基本组件，本课将详细介绍这些组件。

- Oracle 服务器：Oracle 服务器中包含多种文件结构、进程结构和内存结构；但是，处理 SQL 语句时，并非所有这些结构都会用到。某些结构用于提高数据库的性能，确保该数据库在遇到软件或硬件错误时可以恢复，或者执行维护该数据库所需的其它任务。Oracle 服务器包括一个 Oracle 例程和一个 Oracle 数据库。
- Oracle 例程：Oracle 例程是后台进程和内存结构的组合。只有启动例程后，才能访问数据库中的数据。每次启动例程时，会分配系统全局区 (SGA) 并启动 Oracle 后台进程。后台进程代表调用进程执行各种功能。它们把为每个用户运行的多个 Oracle 程序所处理的功能统一起来。后台进程执行输入/输出 (I/O)，并监视其它 Oracle 进程来提高并行性，从而使性能和可靠性更加优越。
- Oracle 数据库：Oracle 数据库包含操作系统文件（也称为数据库文件），这些文件为数据库信息提供了实际的物理存储。数据库文件用于确保数据一致性并能在例程失败时得以恢复。

- 其它关键文件：非数据库文件用于配置例程、验证特权用户以及在磁盘出现故障时恢复数据库。
- 用户进程和服务器进程：执行 SQL 语句时，用户进程和服务器进程是其中涉及的主要进程；但是，其它进程也会有助于服务器完成 SQL 语句的处理。
- 其它进程：还有很多供其它选件使用的其它进程，例如，高级排队 (Advanced Queuing)、实时应用集群 (Real Application Clusters)、共享服务器 (Shared Server) 和高级复制 (Advanced Replication) 等。这些进程将在相应的课程中分别进行讨论。

Oracle 服务器由 Oracle 实例和 Oracle 数据库两大部分组成。

1）Oracle 实例

是一种数据库访问机制，主要由**内存结构**和**进程结构**组成。

内存结构主要包括系统全局区（System Global Area，SGA）、进程全局区（Process Global Area，PGA）等

实例的后台进程有 5 个是必须的

SMON 系统监视器进程　PMON 进程监视器进程　DBWR 数据库书写器

LGWR 日志书写器　CKPT 检查点进程

每个实例只能操作其对应的一个数据库，但一个数据库可以同时被几个实例操作（RAC）

内存结构 P9　进程结构 P12

2）Oracle 数据库

由以下三种操作系统文件组成：

控制文件（control files）数据文件（data files）重做日志文件（redo log files）

oracle 数据库是一个被统一处理的的数据的集合，从物理角度来看包括三类文件数据文件，控制文件，重做日志文件。从逻辑角度来看，oracle 数据库至少包含一个表空间，表空间至少包含一个段，段由区做成，区有块组成。需要注意的是表空间可以包含若干个数据文件，段可以跨同一个表空间的多个数据文件，区只能在同一个数据文件内。

3）其它服务器文件：

初始化参数文件（parameter files）口令文件（password files）归档重做日志文件（archived redo log files）

2. Logmnr 的使用

logmnr 是用于分析日志的工具，主要有以下几个用途：

1. 查明数据库的改变登记：能够用 Logmnr 来分析这些事务,看看究竟发生了些什么事情

2. 找回丢失的数据,当不能利用 flashback 时候,能够利用 Logmnr 工具来找回数据,这个时候,只必需有归档日志即可.

或 ①跟踪数据库的变化：可以离线的跟踪数据库的变化，而不会影响在线系统的性能。

②回退数据库的变化：回退特定的变化数据，减少 point-in-time recovery 的执行。

③优化和扩容计划：可通过分析日志文件中的数据以分析数据增长模式

logmnr 工具包含的过程与视图

dbms_logmnr_d 包包含了：

(1)add_logfile:用来增加/剔除用于分析的日志文件.

(2)start_logmnr:用来开启日志分析,而且在 9i/10g 中,能够开启许多不同的分析选项.

(3)end_logmnr:用来终止分析会话,它将回收 LogMiner 所挪借的内存

与 LogMiner 相关的数据字典:

v$logmnr_dictionary:LogMiner 可能利用的数据字典消息.

v$logmnr_parameters:目前 LogMiner 所设定的参数消息.

v$logmnr_logs:目前用于分析的日志列表.

v$logmnr_contents:日志分析收获.

使用示例 1:

```
SQL> exec
sys.dbms_logmnr.add_logfile(LogFileName=>'c:\oradata\jssweb\arc\20120911_35.arc',options
=>dbms_logmnr.new);
--SQL> exec
sys.dbms_logmnr.add_logfile(LogFileName=>'c:\oradata\jssweb\arc\20120911_36.arc',options
=>dbms_logmnr.addfile);
--开启日志分析
--DBMS_LOGMNR_D.STORE_IN_FLAT_FILE:将数据字典提取到一个平面数据字典文件
--DBMS_LOGMNR_D.STORE_IN_REDO_LOGS:将数据字典提取到重做日志文件
--DBMS_LOGMNR.DICT_FROM_ONLINE_CATALOG:使用当前的数据库的联机数据字典(只
能分析当前数据库的重做日志文件，此时就不需要平面字典文件)
SQL> exec
sys.dbms_logmnr.start_logmnr(options=>sys.dbms_logmnr.dict_from_online_catalog);
--查看分析结果
SQL> select t.scn,t.timestamp,t.seg_owner,t.operation from v$logmnr_contents t where
t.seg_name='MYTESTTAB';
--释放内存
SQL> exec dbms_logmnr.end_logmnr;
```

使用示例 2：

1). 修改初始化参数 SQL> alter system set utl_file_dir='C:\oracle\logmnr\' SCOPE=SPFILE;

2). 重启数据库

3). 提取数据字典文件 SQL> execute
dbms_logmnr_d.build('logmnr_dict.ora','c:\oracle\logmnr\',options=>dbms_logmnr_d.store_in_f
lat_file);

4). 指定 LogMiner 要分析的重做日志文件 SQL> EXECUTE DBMS_LOGMNR.ADD_LOGFILE('C:\oradata\orcl\REDO01.LOG',options=>dbms_logmnr.NEW);
SQL> EXECUTE DBMS_LOGMNR.ADD_LOGFILE('C:\oradata\orcl\REDO02.LOG',options=>dbms_logmnr.ADDFILE); SQL> EXECUTE DBMS_LOGMNR.ADD_LOGFILE('C:\oradata\orcl\REDO03.LOG',options=>dbms_logmnr.ADDFILE); 5). 启动 LogMiner 会话

--options=>dbms_logmnr.NO_ROWID_IN_STMT 取消"ROWID="的内容

--dbms_logmnr.DICT_FROM_ONLINE_CATALOG 只分析当前数据库的重做日志文件

-options=>dbms_logmnr.NO_ROWID_IN_STMT+dbms_logmnr.DICT_FROM_ONLINE_CATALOG

SQL>execute dbms_logmnr.start_logmnr(dictfilename=>'c:\oracle\logmnr\logmnr_dict.ora');

6). 查看结果

SQL> SELECT * FROM V$LOGMNR_LOGFILE;

SQL> select t.scn,t.timestamp,t.seg_owner,t.operation from v$logmnr_contents t where t.seg_name='MYTESTTAB';

7). 结束 LogMiner 会话

SQL> EXECUTE DBMS_LOGMNR.END_LOGMNR;

笔记部分 logmnr 的使用 1 启用数据库补充日志 2 产生数据库字典文件 1）指定数据字典文件的目的地 2）生成数据字典文件 3 产生一个事务 4 添加需要分析的日志文件 5 启动分析 6 查询内容

与 Oracle 服务器的连接

一个连接即称为一个会话，连接可分为以下两种：

1、专用服务器连接

一个用户进程对应创建一个服务器进程，用户进程与服务器进程是一对一的关系。

2、共享服务器连接

多个用户进程同时对应一个服务器进程。

各种不同的连接方式

1、基于主机方式

用户进程与服务器进程运行在同一台计算机相同的操作系统下，用户进程与 Oracle 服务器的通信是通过操作系统内部的进程通信（inter process communication，IPC）机制来建立的。

2、客户端-服务器（client-server）（两层模型）方式

用户进程与 Oracle 服务器的通信是通过网络协议（如 TCP/IP）来完成的

3、客户端-应用服务器-服务器（client-application server-server）（三层模型）方式

用户的个人计算机通过网络与应用服务器或网络服务器进行通信，该应用服务器或网络服务器同样再通过网络与运行数据库的计算机连接。

Oracle 执行 SQL 查询语句的步骤

1、SQL 正文放入共享池（shared pool）的库缓存（library cache）。

2、检查是否有相同的 SQL 正文，没有就进行以下编译处理，否则跳过。

1）语法检查

2）通过数据字典检查表和列的定义

3）对所操作的对象加编译锁，防止编译期间的对象定义被改变

4）检查用户权限

5）生成执行计划

6）将编译后的代码和执行计划放入共享 SQL 区

3、执行

由服务器进程执行 SQL 语句。

4、提取数据

由服务器进程选择所需的数据行，需要时排序（PGA 中），返回给用户进程。

AWR　P241

Automatic Workload Repository (AWR) 收集、处理和维护用于问题诊断的性能统计信息。该数据既存在于数据块中，也存在于内存中。AWR 收集的数据可以通过报告和视图进行查看。

AWR 处理和收集的统计信息包括：

1.确定数据块 segment 访问路径和使用情况的对象统计信息

2.基于数据库活动的时间使用情况的时间模型统计信息，可在 V$SYS_TIME_MODEL 和 V$SESS_TIME_MODEL 视图中查看

3.V$SYSSTAT 和 V$SESSTAT 视图中收集的一些 sytem 和 session 的统计信息

4.按照 elapsed time 和 CPU time 等条件在系统上筛选出的产生较高负载的 SQL 语句

5.ASH 统计信息——最近的 session 活动的历史记录

数据库默认情况下已启用 AWR 收集统计信息，它 STATISTICS_LEVEL 初始化参数来控制。STATISTICS_LEVEL 参数必须设置为 TYPICAL 或 ALL 才能启用 AWR 统计信息收集。默认的设置为 TYPICAL。将 STATISTICS_LEVEL 设置为 BASIC 将禁用许多 Oracle Database 功能，包括 AWR，所以不推荐这么设置。当 STATISTICS_LEVEL 设置为 BASIC 时，仍然可以使用 DBMS_WORKLOAD_REPOSITORY 包手动捕获 AWR 统计信息。但是许多在内存中收集的系统统计信息，如 segment 统计信息和 memory advisor 信息都将被禁用，这种情况下的手动快照捕获的统计信息可能不是完整的。

1 Snapshot

快照是 ADDM 用于性能比较的特定时期内的历史数据集合。在 11g 中，Oracle Database 每小时会自动生成性能数据的快照，并将这些统计信息在工作负载信息库中保留 8 天。您也可以手动创建快照，其实没必要这么做。快照间隔内的统计信息由 Automatic Database Diagnostic Monitor (ADDM) 进行分析。

AWR 通过比较各个快照之间的差异，根据对系统负载的影响来确定要捕获的 SQL 语句，随着时间的发展，必须捕获的 SQL 语句将逐渐减少。

2 Baseline

Baseline 是指一个特定时间段内的性能数据，保留这些数据是为了在性能问题产生时与其他类似的工作负载时间段进行比较。Baseline 中包含的快照将从自动 AWR 清理进程中排除，并无限期的保留。在 Oracle Database 中存在多种类型的 baseline；

Fixed Baseline：fixed baseline 表示的是您指定的一个固定的、连续的时间段。在创建 fixed baseline 之前，请认真考虑您选作 baseline 的时间段，因为该 baseline 应该代表系统处于良好的性能下运行。您可以在将来将该 baseline 与在性能较差的时间段捕获的其他 baseline 或 snapshot 进行比较分析。

Moving Window Baseline：表示的是 AWR 保留期内存在的所有 AWR 数据。在使用自适应阈值时，它非常有用，因为数据库可以使用整个 AWR 保留期内的 AWR 数据来计算指标值。

Oracle Database 会自动维护系统定义的 moving window baseline。系统定义的 moving window baseline 的默认窗口大小就是当前的 AWR 保留期，即默认为 8 天。如果您打算使用自适应阈值，请考虑使用更长的移动窗口——如 30 天，以便精确地计算阈值。您可以重新调整 moving window baseline，将移动窗口的大小调整为小于或等于 AWR 的保留天数。因此，要增加移动窗口的大小，必须要先增加相应的 AWR 保留期限。

Baseline Template：您可以使用 baseline template 创建将来某个连续时间段的 baseline。Oracle 中有两种 baseline 模板：single 和 repeating

利用 single baseline template，您可以为将来某个单独的连续时间段创建 baseline。该技术在某些情况下非常有用。例如，如果您想捕获下周计划的系统测试期间的 AWR 数据，您可以创建一个 single baseline template 来自动捕获测试发生的时间段的统计数据。

利用 repeating baseline template ，可以根据重复的时间计划创建和删除 baseline。当您希望 Oracle Database 自动持续地捕获连续时间段的统计数据时，这非常有用。例如，您可能需要在长达一个月内捕获每周一早上的 AWR 数据。在这种情况下，您可以创建一个 repeating baseline template ，以在每周一自动创建 baseline，在指定的过期期限内自动删除过时的 baseline。

AWR 操作

3.1. 查看当前的 AWR 保存策略

```
SQL> col SNAP_INTERVAL format a20
SQL> col RETENTION format a20
SQL> select * from dba_hist_wr_control;
      DBID SNAP_INTERVAL          RETENTION                 TOPNSQL
---------- -------------------- -------------------- ----------
262089084 +00000 01:00:00.0       +00007 00:00:00.0       DEFAULT
```

以上结果表示,每小时产生一个 SNAPSHOT，保留 7 天。

3.2. 调整 AWR 配置

AWR 配置都是通过 dbms_workload_repository 包进行配置。

3.2.1 调整 AWR 产生 snapshot 的频率和保留策略，如将收集间隔时间改为 30 分钟一次。并且保留 5 天时间（单位都是分钟）：

```
SQL> exec dbms_workload_repository.modify_snapshot_settings(interval=>30, retention=>5*24*60);
```

3.2.2 关闭 AWR，把 interval 设为 0 则关闭自动捕捉快照

```
SQL> exec dbms_workload_repository.modify_snapshot_settings(interval=>0);
```

3.2.3 手工创建一个快照

```
SQL> exec DBMS_WORKLOAD_REPOSITORY.CREATE_SNAPSHOT();
```

3.2.4 查看快照

```
SQL> select * from sys.wrh$_active_session_history
```

3.2.5 手工删除指定范围的快照

```
SQL> exec DBMS_WORKLOAD_REPOSITORY.DROP_SNAPSHOT_RANGE(low_snap_id => 973, high_snap_id => 999, dbid => 262089084);
```

3.2.6 创建 baseline，保存这些数据用于将来分析和比较

```
SQL> exec dbms_workload_repository.create_baseline(start_snap_id => 1003, end_snap_id => 1013, 'apply_interest_1');
```

3.2.7 删除 baseline

```
SQL> exec DBMS_WORKLOAD_REPOSITORY.DROP_BASELINE(baseline_name => 'apply_interest_1', cascade => FALSE);
```

3.2.8 将 AWR 数据导出并迁移到其它数据库以便于以后分析

```
SQL> exec DBMS_SWRF_INTERNAL.AWR_EXTRACT(dmpfile => 'awr_data.dmp', mpdir => 'DIR_BDUMP', bid => 1003, eid => 1013);
```

3.2.9 迁移 AWR 数据文件到其他数据库

SQL> exec DBMS_SWRF_INTERNAL.AWR_LOAD(SCHNAME => 'AWR_TEST', dmpfile => 'awr_data.dmp', dmpdir => 'DIR_BDUMP');

把 AWR 数据转移到 SYS 模式中：

SQL> exec DBMS_SWRF_INTERNAL.MOVE_TO_AWR (SCHNAME => 'TEST');

检查点的作用：(1)同步所有的数据文件 (2)同步所有的控制文件 (3)发送信号通知

回滚段 作用 Rollback 1.读一致性 2.回退 3.闪回恢复

例程管理 1.初始化参数文件 1)初始化参数文件：文件中的条目专用于要启动的例程；两种类型参数：显式/文件中有一个条目，隐式/文件中没有条目，假定取 or 缺省值；可存在多个初始化参数文件；两种类型的文件：静态参数文件 pfile(文本文件，使用操作系统编辑器进行修改，手动修改，所做更改在下次启动时生效，仅在例程启动过程中打开，只读，缺省位置:$ORACLE_HOME/dbs，initSID.ora)/永久参数文件 spfile(二进制文件，由 oracle 服务器维护，始终驻留在服务器端，所做更改永久有效，不受关闭和启动的影响，自行调节参数值，使恢复管理器能够备份初始化参数文件) 2）查看初始化参数 3）修改初始化参数 4）将初始化参数还原成默认值 5）修复出错的初始化参数 6）根据 spfile/pfile 创建 pfile/spfile 7）例程启动时选择初始化参数文件的顺序(starup) Spfile<sid>.ora/缺省 SPFILE/initSID.ora/缺省 PFILE 2.启动数据库 1）启动例程 MOMOUNT：从 ORACLE_HOME/dbs 读取初始化文件/分配 SGA/启动后台进程/打开 alertSID.log 文件和跟踪文件 2）加载数据库 MOUNT ：使数据库与以前启动的例程关联/定位并打开参数文件中指定的控制文件/读取控制文件以获取数据文件和重做日志文件的名称和状态。3）打开数据库 OPEN：打开联机数据文件/打开联机重做日志文件

视图的作用 1）简化查询 2）收集感兴趣的数据 3）屏蔽敏感数据 4）简化权限管理 P178 P152

手工创建数据库(sales) 1.创建 windows 服务(instance):oradim -new –sid 2 将 sales 设为当前例程 :set oracle_sid=sales 3 创建/编辑初始化参数文件:create pfile from spfile; 4 根据初始化参数创建相应的目录结构 :startup nomount 5 执行创建数据库的命令 :create database sales
datafile 'F:\oracle\oradata\sales\system01.dbf' size 350m
sysaux datafile 'F:\oracle\oradata\sales\sysaux01.dbf' size 350m
undo tablespace undotbs1 datafile 'F:\oracle\oradata\sales\undotbs01.dbf' size 50m
default temporary tablespace temp tempfile 'F:\oracle\oradata\sales\temp01.dbf' size 30m
logfile group 1 ('F:\oracle\oradata\sales\redo01.log') size 10m,
group 2 ('F:\oracle\oradata\sales\redo02.log') size 10m, group 3 ('F:\oracle\oradata\sales\redo03.log') size 10m;
6 创建表 :create table 7 创建数据字典视图 desc :脚本 dbhome\rdbms\.sql :@?\rdbms\admin\catalog
8 创建 oracle 内部包 :catproc.sql :@?\rdbms\admin\catproc 9 创建 spfile :create spfile from pfile
10 创建 Scott 方案:@?\rdbms\admin\scott conn as sys alter user scott identified by tiger
产品用户概要文件11 加载产品用户概要文件信息 :conn system/manager :@?\sqlplus\admin\pupbld
12 配置监听器(服务端)和服务名(客户端) :lsnrctl stop/start conn /@sales sysdba
13 配置 DBConsole(EM\OEM) 1)准备用户 :sys :dbsnmp 解锁修改密码
:sysman 创建用户 >desc dba_users >alter user dbsnmp account unlock identified by admin
2)创建口令验证文件:orapwd file=F:\oracle\product\11.2.0\dbhome_3\database\PWDsales.ora entries=4
3)运行 emca :emca -config dbcontrol db -repos create
三个必须创建的表空间 system sysaux undo

管理控制文件：1.控制文件的作用 select name from v$controlfile 2.oradebug 转储成文本文件
oradebug dump controlf 3 :error oradebug setmypid oradebug dump controlf 3
>查看文件路径 :oradebug tracefile_name 3.控制文件的内容 4.控制文件复用(多个控制文件在不同地方)
修改初始化参数 >show parameter control_files :alter system set control_files='F:\ORACLE\ORADATA\SALES\CONTROL01.CTL','F:\ORACLE\FLASH_RECOVERY_AREA\SALES\CONTROL02.CTL','G:\CONTROL03.CTL' scope=spfile; >show parameter control_files :shutdown immediate

>复制粘贴 >启动！ 5.创建控制文件>alter database backup controlfile to trace as 'G:\bak.txt'; :shutdown immediate >删除数据库控制文件 control01.ctl,control02.ctl,control03.ctl :startup nomount :alter database mount >复制粘贴 :alter database open 6.组里多个成员是镜像关系 循环工作 :select group#,sequence#,status from v$log; :select group#,members from v$logfile;

管理日志文件：1.创建 1)添加日志文件组:alter database add logfile group 4 ('F:\oracle\oradata\sales\redo04.log') size 10m; 2)添加文件组成员:alter database add logfile member 'F:\oracle\oradata\sales\redo04a.log' to group 4; 2.修改 清除日志文件 :alter database clear logfile 3. 删除 1)删除文件组成员(操作系统并未删掉文件) :alter database drop logfile member 'F:\oracle\oradata\sales\redo04a.log';

2)删除文件组 :alter system switch :alter system checkpoint; :alter database drop logfile group 3;

4.OMF oracle 自己管理 :show parameter db_create_file_dest >新建目录 OMF :alter system set db_create_online_log_dest_1='D:\OMF' :select group#,members from v$logfile; :alter database add logfile; :alter database adrop logfile grop 3; 5.设置数据库归档模式 :archive log llist :alter database (non)archivelog;(必须在 mount 阶段做) >shutdown->startup nomount->alter database mount 6.设置归档目的地 :show parameter log_archive; 创建一个目录 :alter system set log_archive_dest_1='location=D:\' :alter syetem system archive log curren 7.手工归档 8.移动/重命名日志文件:shutdown immediate :startup mount :select member from v$logfile; :alter database rename '原文件路径' to '现文件路径:alter database open; 9.修复日志文件 1)非当前日志文件 :select group#,sequence#,status from v$log; :select group#,members from v$logfile; :startup mount :alter database clear logfile group 1; 2)当前日志文件 :update scott.emp set sal=900 where empno=7698; :select group#,status from v$log; 3)删除文件 :startup mount :recover database until cancel; :alter database open resetlogs;

RWAN 整库备份 P377：1）配置 spfile 和 controlfile 自动备份 2）配置数据文件和归档日志文件的备份目的地 3）进行整库备份 4）再插入数据 5）复制日志文件 6）摧毁数据库 7）恢复数据库 1 oradim 2 创建相应的目录结构 3 startup nomount 4 restore spfile 5 restore controlfile

数据库类型 1 目标数据库 2 目录数据库 3 辅助数据库

备份类型 1 用户管理的备份 1）冷备份（1）文件列表（2）shutdown2）热备份（1）归档模式（2）备份模式 2 服务器管理的备份

审核 1 强制审核（默认审核）——警告日志文件 2 标准数据库审核 1）启用审核——修改初始化参数 2）制定审核选项（1）用户审核（权限审核）（2）对象审核（3）语句审核 系统权限之前是验证，强制审核之前是授权