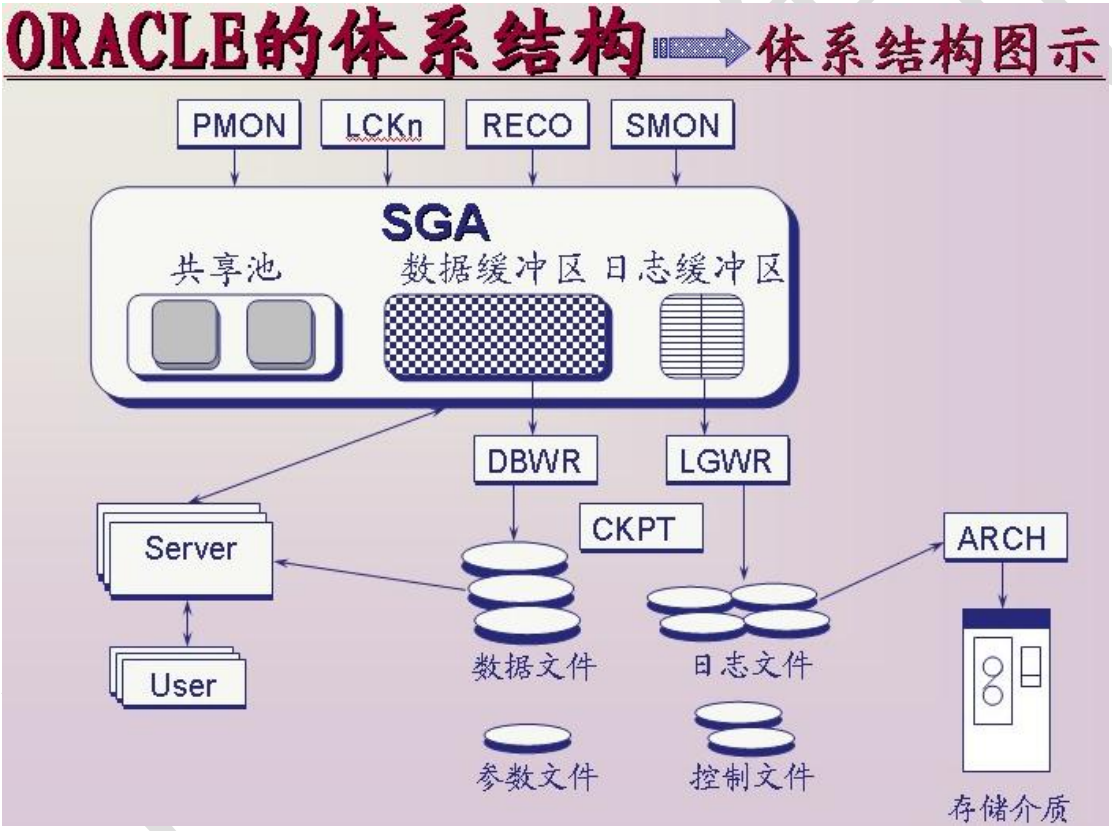
1. ORACLE 体系结构



① 物理结构

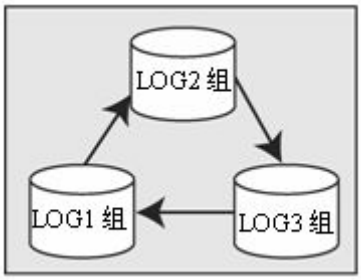
✓ 数据文件

数据文件用来存储数据库中的全部数据，如数据库表中的数据和索引数据。通常为后缀名为.dbf 格式的文件； 一个数据库实例可以包含多个数据文件，一个数据文件只能存储一个表空间。(添加网卡：devcon.exe install %windir%\inf\netloop.inf *msloop)

✓ 日志文件

日志文件，用于记录数据库所做的全部变更（如增加、删除、修改），以便在系统发生故障时，用它对数据库进行恢复。名字通常为 Log*.dbf 格式

根据在事务信息将被覆盖时，是否应该将文件归档，数据库分为以下两种归档模式：ARCHIVELOG（归档日志）或 NOARCHIVELOG（非归档日志）模式。



比如：如果用户删除某个表时，重做日志信息会写入日志文件，日志文件的生成是通过递归方式生成数据恢复和表恢复 sql

oracle 的日志文件至少有两个，当第一个写满了，就会接着写第二个，第二写满后，如果是归档模式就会要求你提供路径保存日志文件，如果是非归档模式则会直接覆盖第一个日志文件

检查点的作用：脏数据块写入相应的数据文件中，并且同步数据文件头和控制文件，保证数据库的一致。 1、重做日志切换 2、手动执行系统命令 3、数据库正常 shutdown
用户有数据修改时，后台进程同时纪录：数据库缓冲区数据和重做日志缓冲区，当检查点到达后数据缓冲区数据被写入数据，检点之前的修改都是内存操作；当用户修改后执行 commit 命令时，重做日志缓冲区数据被写入重做日志
磁盘陈列概念： 不能保护操作员错误导致的数据丢失（RAID-5）

✓ 控制文件

每个 Oracle 数据库都有相应的控制文件，用于打开、存取数据库。它们是较小的二进制文件，其中记录了数据库的物理结构。名字通常为 Ctr*.ctl 格式
每个数据库必须至少拥有一个控制文件。一个数据库也可以同时拥有多个控制文件，但是一个控制文件只能属于一个数据库(数据库创建后默认创建三个控制文件，内容一样，主要是备份作用)

重要提示：
包含维护和验证数据库完整性的必要信息、例如，控制文件用于识别数据文件和重做日志文件，
控制文件内容
数据库名
表空间信息
所有数据文件的名字和位置
所有 redo 日志文件的名字和位置
当前的日志序列号
检查点信息
关于 redo 日志和归档的当前状态信息

控制文件的使用过程

控制文件把 Oracle 引导到数据库文件的其它部分。启动一个实例时，Oracle 从参数文件中读取控制文件的名字和位置。安装数据库时，Oracle 打开控制文件。最终打开数据库时，Oracle 从控制文件中读取数据文件的列表并打开其中的每个文件

✓ 配置文件

配置文件是一个 ASCII 文本文件，记录 Oracle 数据库运行时的一些重要参数。名字通常为 initsid*.ora 格式，如：initCIMS.ora，SID 相当于它所控制的数据库的标识符。每个 Oracle 数据库和实例都有它自己惟一的 init.ora 文件。

Oracle9i 新引入一个服务器参数文件（SPFILE），一个服务器参数文件（SPFILE）可以被认为是在 Oracle 数据库服务器端的初始化参数文件。存储在一个服务器参数文件的初始化参数是永久的，它提供了由 Oracle 数据库服务器自我调节的一个基础。服务器参数文件是二进制文件，不能使用一个文本编辑器浏览或编辑。Oracle 提供了浏览和查看相关参数的另外接口。
initdw.ora、listener.ora(配置监听位置)、sqlnet.ora、init.ora（数据库配置参数）

增加说明：
一个实例由一组数据文件、控制文件、日志文件和一组后台进程组成 ------ 基本概念不清
数据库最基本的由 数据文件、控制文件、日志文件、密码文件、初始化参数文件 构成
实例是 内存和进程的资源的集合，实例是用来管理数据库的， SGA + PGA+(lgwr,dbwr,smon,pmon,ckpt.....)
表空间是一个逻辑的单位，由一个或多个数据文件构成，一个数据文件只能属于一个表空间
extent 是在物理上连续的 block 构成，一个 extent 只在一个数据文件内
一个表，非分区表，其存储就是一个段，一个段只能在一个表空间里面，一个段可由一个或者多个 extent 构成
对于分区表，一个表可以由多个段构成，这些段可以分布在不同的表空间中

一个实例由一组数据文件、控制文件、日志文件和一组后台进程组成
表空间是逻辑存储空间，用于存储 Oracle 数据对象，如：用户、表、索引、视图等
数据文件是物理存储文件，几乎所有 Oracle 数据对象实际都存储在数据文件中
表空间和数据文件的关系：表空间有 1 个或多个数据文件组成，数据文件只能属于一个表空间。
另外有一个重要的文件：控制文件，主要存储数据库参数数据，如数据库参数、表空间信息等。

② 逻辑结构

✓ 表空间：System 表空间、回滚表空间、临时表空间。
根据需要分离数据，实现分布式存储，如：用户数据同回退数据分开、分布式管理（可以将表空间设为脱机进行备份恢复）、可以自定义专用表空间便于管理（表空间只读、表空间大小管理）。
一个表空间可以对应多个存储文件。
✓ 段：数据段（存储对象数据 表）、索引段（）、临时段（记录新的操作，增加性能如 查询、排序）、回退段（数据回滚、恢复）
✓ 区：数据一次性预留的一个较大的存储空间，由许多数据块组成存储空间。

- ✓ 数据库块： 逻辑块或 ORACLE 块。PCTFREE、PCTUSED
- ✓ 模式对象：指 表、过程、同义词、触发器、序列、索引对象

重要内容：

③ 内存结构

- ✓ 系统全局区：系统全局区（SGA，System Global Area.）

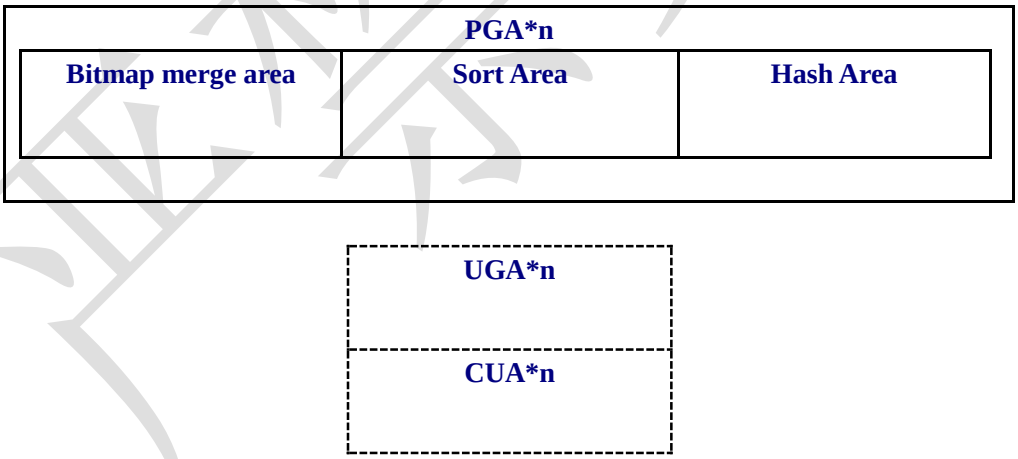
Oracle 为一个实例分配的一组共享内存缓冲区，保存着 Oracle 系统与所有数据库用户的共享信息，包括数据维护、SQL 语句分析，重做日志管理等。 包括：
数据缓冲区、字典缓冲区、重做日志缓冲区、共享 SQL 池、JAVA 池、多缓冲池

SGA

Share Pool	**Buffer Cache**	**Redo Log Buffer**
Java Pool	**Stream Pool(10g)**	**Large Pool**

- ✓ 程序全局区：程序全局区 PGA（Program Global Area，PGA）

单个 Oracle 进程使用的内存区域，不属于实例的内存结构。它含有单个进程工作时需要的数据和控制信息，PGA 是非共享的，只有服务进程本身才能够访问它自己的 PGA 区



- ✓ 排序区：存在于请求排序的用户进程的内存中

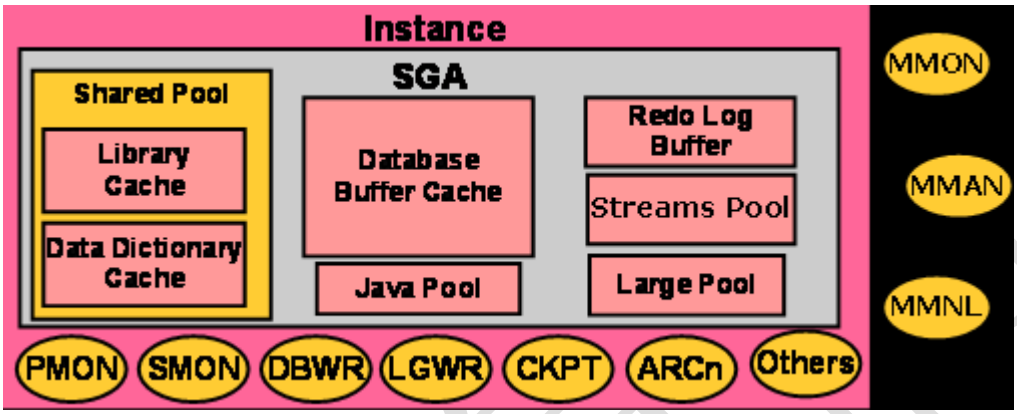
该区域受排序数据量的大小可增大，但是会受 SORT_AREA_SIZE 参数限制

- ✓ 软件代码区：存储正在执行的或可以执行的程序代码

软件代码区是只读，可安装成共享或非共享。Oracle 系统程序是共享的，多个 Oracle 用户可存取它，用户程序可以共享也可以不共享

④ 数据库实例与进程

- ✓ Oracle 数据库实例：指软件系统中用来访问数据库文件集的存储结构以及后台进程的集合，它是存取和控制数据库的软件机制（数据库：指物理上的数据库文件或逻辑上的数据库结构）
- ✓ 数据库进程：在 ORACLE 体系中指：用户进程、服务器进程和后台进程的一个集合



ORACLE 实例简图

⑤ 数据字典

- ✓ 只读类型的表和视图组成，保存着关于数据库系统本身以及其中存储的所有对象的基本信息，如：数据库安全信息、审计信息、等等
- ✓ 用途：对于 Oracle 系统本身而言，当数据库实例运行时，会需要使用数据字典基础表中的信息。Oracle 从基础表中读取信息，来判断用户要求访问的对象是否存在。同时，当用户对数据库结构、对象结构做出修改时，Oracle 向基础表中写入相应的修改信息

2. ORACLE 9i 与 10i 数据库各版本之间的差别

① 10i 内存占用多

② 管理习惯改变

- ✓ 采用 web 管理 EM，增加了自动化程度，不少参数不需要特别维护

③ 闪回恢复区

- ✓ 将所有恢复相关的文件，比如 flashback log,archive log,backup set 等，放到这个区域集中管理；闪回查询：在提供整个数据库回退到过去某个时刻的能力，这是通过引入一种新的 flashback log 实现的。flashback log 有点类似 redo log，只不过 redo log 将数据库往前滚，flashback log 则将数据库往后滚。为了保存管理和备份恢复相关的文件，oracle10g

提供了一个叫做闪回恢复区(Flashback recovery area)的新特性，可以

④ 查看闪回区大小

show parameter db_recovery_file_dest

⑤ 修改闪回区大小

ALTER SYSTEM SET db_recovery_file_dest_size=4g scope=both;

⑥ 如何删除闪回区的归档日志

⑦ 闪回举例

- ✓ // 表数据闪回需要,设置表行可移动

```
SQL>alter table test_stud1 enable row movement;
SQL> flashback table GEOSTAR.GEOSTAR_OBJECT_CLASSES$ to timestamp to_timestamp('2010-05-03 13:00:00','yyyy-mm-dd hh24:mi:ss');
```

- ✓ // 显示垃圾堆

```
SQL> show recyclebin
SQL> show recyclebin
    Y62     BIN$ha9iRE3/tvHgQKjAsAE2iA==$0 TABLE       2010-05-03:18:52:06
```

- ✓ // 对表对象删除的闪回

```
    SQL>flashback table test_drop to before drop
或者
    SQL>lashback table "BIN$b+XkkO1RS5K10uKo9BfmuA==$0" to before drop;
```

（）

- ✓ 管理回收站

```
    清除回收站中的单个表：purge table test_drop
    清除整个回收站：purge recyclebin
    清除不同的对象回收站：purge user_recyclebin 或 purge dba_recyclebin
```

- ✓ 确认删除一个表

```
    drop table test_drop purge;
```

- ✓

2.1 i/g 的含义

- ✓ I： 表示 ORACLE 产品对企业内部互联网和因特网的支持
- ✓ G：表示 ORACLE 产品对网格的支持

3. ORACLE 内置程序介绍和使用方法

① SMON 系统监控进程

该进程实例启动时执行实例恢复，还负责清理不再使用的临时段。在具有并行服务器选项的环境下，SMON 对有故障 CPU 或实例进行实例恢复。SMON 进程有规律地被呼醒，检查是否需要，或者其它进程发现需要时可以被调用。

② PMON 进程监控进程

该进程在用户进程出现故障时执行进程恢复，负责清理内存储区和释放该进程所使用的资源。例：它要重置活动事务表的状态，释放封锁，将该故障的进程的 ID 从活动进程表中移去。PMON 还周期地检查调度进程（DISPATCHER）和服务器进程的状态，如果已死，则重新启动（不包括有意删除的进程）。

③ DBWR 数据库写进程

该进程执行将缓冲区写入数据文件，是负责缓冲存储区管理的一个 ORACLE 后台进程。当缓冲区中的一缓冲区被修改，它被标志为“弄脏”，DBWR 的主要任务是将“弄脏”的缓冲区写入磁盘，使缓冲区保持“干净”。由于缓冲存储区的缓冲区填入数据库或被用户进程弄脏，未用的缓冲区的数目减少。当未用的缓冲区下降到很少，以致用户进程要从磁盘读入块到内存存储区时无法找到未用的缓冲区时，DBWR 将管理缓冲存储区，使用户进程总可得到未用的缓冲区。
ORACLE 采用 LRU（LEAST RECENTLY USED）算法（最近最少使用算法）保持内存中的数据块是最近使用的，使 I/O 最小。在下列情况预示 DBWR 要将弄脏的缓冲区写入磁盘
当一个服务器进程将一缓冲区移入“弄脏”表，该弄脏表达到临界长度时，该服务进程将通知 DBWR 进行写。该临界长度是为参数 DB-BLOCK-WRITE-BATCH 的值的一半。
当一个服务器进程在 LRU 表中查找 DB-BLOCK-MAX-SCAN-CNT 缓冲区时，没有查到未用的缓冲区，它停止查找并通知 DBWR 进行写。出现超时（每次 3 秒），DBWR 将通知本身。当出现检查点时，LGWR 将通知 DBWR.在前两种情况下，DBWR 将弄脏表中的块写入磁盘，每次可写的块数由初始化参数 DB-BLOCK-WRITE-BATCH 所指定。如果弄脏表中没有该参数指定块数的缓冲区，DBWR 从 LUR 表中查找另外一个弄脏缓冲区。

④ LGWR 日志文件写进程

⑤ ARCH 归档进程

该进程将已填满的在线日志文件拷贝到指定的存储设备。当日志是为 ARCHIVELOG 使用方式、并可自动地归档时 ARCH 进程才存在

⑥ RECO 恢复进程

该进程是在具有分布式选项时所使用的一个进程，自动地解决在分布式事务中的故障。一个结点 RECO 后台进程自动地连接到包含有悬而未决的分布式事务的其它数据库中，RECO 自动地解决所有的悬而不决的事务。任何相应于已处理的悬而不决的事务的行将从每一个数据库的悬挂事务表中删去

⑦ LCKN 封锁进程

是在具有并行服务器选件环境下使用，可多至 10 个进程（LCK0，LCK1……，LCK9），用于实例间的封锁

⑧ 服务进程

- ✓ Net Manager：
  管理客户端连接进程。提供分布式异构平台企业级连接解决方案，是 NET SERVER 组件之一，提供客户端到服务器的网络会话能力，建立网络会话后，负责数据传递，ORACLE NET 位于网络中每台计算机；
- ✓ Net Configuration Assistant：
  配置向导，可以配置数据库监听器、网络服务名、命名方法、目录使用
- ✓ Wallet Manager
- ✓ Enterprise Security Manager
- ✓ Policy Manager

4. ORACLE 开发

① oci 编程（Oracle Call Interface）

- ✓ 介绍

OCI 是由头文件和库函数组成的一套 ORACLE 数据库应用编程接口。

- ✓ OCI 代码分析
  - ✧ 定义操作需要的变量

```
// 定义需要的变量
OCIEnv *m_envhp;              // 管理ORACLE环境变量，是整个操作的基础
OCIError * m_errhp;           // 管理错误信息
OCIServer *m_srvhp;           // 服务句柄， 用来管理连接ORACLE的SID
OCISvcCtx * m_svchp;          // 服务环境句柄， 用来管理数据库会话
OCIStmt *m_stmthp;            // 语句句柄 ， 操作 sql 时需要用到
```

  - ✧ 初始化 OCI 环境

```
// 初始化环境
if (OCIInitialize((ub4)OCI_DEFAULT, (dvoid *)0,(dvoid * (*)(dvoid *, size_t)) 0,
      (dvoid * (*)(dvoid *, dvoid *, size_t))0,(void (*)(dvoid *, dvoid *)) 0 ))
```

```
{
    printf("init OCI entironment failed! error msg:%s\n", GetError(m_errhp));
    return 0;
}

if (OCIEnvInit((OCIEnv**)&m_envhp,OCI_DEFAULT, (size_t)0,(dvoid**)0))
{
    printf("init OCI entironment failed! error msg:%s\n", GetError(m_errhp));
    return 0;
}
// 简要说明：
// 其中m_envhp是输出参数，是一个指向OCI环境的句柄指针
// OCI_DEFAULT是OCI环境的初始化模式
// size_t 是为用户分配的内存数量
// dvoid** 类型变量指向用户的内存区域，该区域大于等于size_t类型变量
```

✧ 申请环境句柄

```
// 申请句柄
OCIHandleAlloc((dvoid*) m_envhp, (dvoid **)&m_errhp, OCI_HTYPE_ERROR,
(size_t)0,(dvoid**)0);
OCIHandleAlloc((dvoid*) m_envhp, (dvoid **)&m_srvhp, OCI_HTYPE_SERVER,
(size_t)0,(dvoid**)0);
OCIHandleAlloc((dvoid*) m_envhp, (dvoid **)&m_svchp, OCI_HTYPE_SVCCTX,
(size_t)0,(dvoid**)0);
OCIHandleAlloc((dvoid*) m_envhp, (dvoid **)&m_stmthp, OCI_HTYPE_STMT,
(size_t)0,(dvoid**)0);

// 说明
// m_errhp 新申请的句柄，m_envhp 为他的父环境句柄
// OCI_HTYPE_ERROR为句柄类型
// OCI_HTYPE_ERROR  错误句柄
// OCI_HTYPE_SERVER 服务句柄
// OCI_HTYPE_SVCCTX 服务环境句柄
// OCI_HTYPE_STMT   语句句柄
// 存储在句柄中的数据称为句柄属性，可以通过调用OCIAttrGet和OCIAttrSet来操作
```

✧ 连接到 ORACLE 实例

```
// 连接服务器
// 建立同服务器的连接
char* dbname="gis";
if (OCIServerAttach(m_srvhp,m_errhp,(text*)dbname,strlen(dbname),OCI_DEFAULT))
{
    printf("open server failed! error msg:%s\n", GetError(m_errhp));
    return 0;
}
```

```
// 说明
// text * 为空表示连接默认数据库
// OCI_DEFAULT 表示操作模式为阻塞方式，只有当 OCI 调用完毕，控制权才回到客户端
```

✧ 登陆数据库

```
// 建立数据库会话
if (OCILogon(m_envhp,m_errhp,&m_svchp,(text*)user,strlen(user),
     (text*) passwd,strlen(passwd), (text*)dbname ,strlen(dbname) ))
{
     printf("login database failed! error msg:%s\n", GetError(m_errhp));
     return 0;
}
```

✧ 准备查询命令

```
// 执行SQL
// 定义
char* sqlcommand = "select * from project" ;// where projectid=1000";

//准备
if (OCIStmtPrepare(m_stmthp, m_errhp, (text*)sqlcommand,
     (ub4)strlen(sqlcommand),(ub4)OCI_NTV_SYNTAX, (ub4)OCI_DEFAULT))
{
     printf("prepare failed! error msg:%s\n", GetError(m_errhp));
     return 0;
}
```

✧ 邦定输出到变量

```
//定义输出变量
OCIDefine *defnp0 = (OCIDefine*) 0 ;

//邦定变量
char szProjectid[32];
//邦定第一个字段PROJECTID
if (OCIDefineByPos( m_stmthp, &defnp0 , m_errhp, 1, (dvoid*) szProjectid,
     sizeof(szProjectid), SQLT_STR, (dvoid *)0, (ub2*)0,(ub2*)0,OCI_DEFAULT))
{
     printf("attach param failed! error msg:%s\n", GetError(m_errhp));
     return 0;
}

char szProjectName[32];
//邦定PROJECTＮＡＭＥ;
if (OCIDefineByPos( m_stmthp, &defnp0 , m_errhp, 2, (dvoid*) szProjectName,
     sizeof(szProjectName), SQLT_STR, (dvoid *)0, (ub2*)0,(ub2*)0,OCI_DEFAULT))
{
     printf("attach param failed! error msg:%s\n", GetError(m_errhp));
     return 0;
```

```
}
```

✧ 执行查询

```
// 执行sql
if (OCIStmtExecute(m_svchp, m_stmthp, m_errhp, (ub4)0, (ub4)0,
    (OCISnapshot*) NULL,(OCISnapshot*)NULL, (ub4)OCI_DEFAULT))
{
    printf("sql execute failed! error msg:%s\n", GetError(m_errhp));
    return 0;
}
```

✧ 提取查询结果

```
// 提取查询结果
sword swResult;
FILE *fp;
fp=fopen("c:\\result.txt","w");

while((swResult=OCIStmtFetch(m_stmthp,
m_errhp,1,OCI_FETCH_NEXT,OCI_DEFAULT)) != OCI_NO_DATA)
{
    fprintf(fp,"%s|%s\n",szProjectid,szProjectName);
}
// 关闭打开的文件
if (fp)
    fclose(fp);
```

✧ 清理现场

```
// 结束会话，断开数据库连接
OCILogoff(m_svchp, m_errhp);

// 断开与数据源的连接
OCIServerDetach(m_srvhp, m_errhp, OCI_DEFAULT);

// 释放句柄
OCIHandleFree((dvoid*) m_stmthp, OCI_HTYPE_STMT) ;
OCIHandleFree((dvoid*) m_svchp, OCI_HTYPE_STMT) ;
OCIHandleFree((dvoid*) m_srvhp, OCI_HTYPE_STMT) ;
OCIHandleFree((dvoid*) m_errhp, OCI_HTYPE_STMT) ;
```

✧ 注意

1、 以上函数的使用，可以参见以下文档，都有讲

2、 其他高级应用，大家有兴趣可以自行研究

✓ 函数说明

Oracle常用的OCI函数.doc

✓ OCI DEMO 工程（VC7+ORA10g 测试成功）

② sqlplus

③ proc*c 编程

✓ 介绍

PRO*C 是 ORACLE 提供的专用开发工具，以 c 为宿主语言，能在 c 代码中嵌入 sql 语句。

✓ 开发环境要求

✧ C 编译器

✧ PRO*C 编译器

✧ 工程包含路径（以下是我的 oracle 配置）

D:\oracle\product\10.2.0\client_1\precomp\public

✓ 编码分析分析

✧ 定义通信区域

```
#include "sqlca.h"
#include "sqlcpr.h"
// 声明宿主变量
EXEC SQL BEGIN DECLARE SECTION;
char szProjectId[16];
char varProjectName[16];
varchar vc_user[0];
EXEC SQL END DECLARE SECTION;
```

✧ 连接数据库

```
strcpy((char*)vc_user.arr,"caojj/caojj123@gis");
vc_user.len = strlen("caojj");
// 连接数据库
EXEC SQL CONNECT :vc_user;

//判断登陆是否成功
if (sqlca.sqlcode == 0)
{
   printf("connect failed!\n");
   return 0;
}
```

✧ 执行查询功能

```
// 实现查询功能， 并将查询结果放入游标
EXEC SQL DECLARE cur_emp CURSOR FOR SELECT projectid,projectname FROM
project;
```

- ✧ 获取数据

```
EXEC SQL OPEN cur_emp;
while()
{
    EXEC SQL FETCH cur_emp INTO :szProjectId, :varProjectName;
    if( sqlca.sqlcode == 0)
    {
        break;
    }
    printf("projectid=%s projectname=%s\n",szProjectId,varProjectName);
}
```

- ✧ 关闭游标

```
EXEC SQL CLOSE cur_emp;
```

✓ 预编译

- ✧ 预编译命令

proc iname=procTest1.cpp oname=procTest.cpp INCLUDE=path CODE=ANSI_C CPP_SUFFIX=cpp SQLCHECK=SEMANTICS USERID=caojj/caojj123@gis

具体请参见 proc 参数说明：

✓ 生成可执行文件

将生成的 procTest.cpp 文件，用 vc 编译即可。

5. ORACLE 不同的连接方式

① ADO.NET

✓ ORACLECLIENT 命名空间结构（结构同 SQLSERVER 空间一致）：

OracleConnectio： 打开 ORACLE 的连接

OracleCommand： 执行的 sql 语句或存储过程

OracleDataReader：读取数据

OracleDataAdapter：ORACLE 数据库和本地 DataSet 的桥梁

OracleParameter： OracleCommand 的参数， sql 语句和存储过程的参数

OracleType： Oracle 数据类型和结构的枚举

OracleParameterCollection： OracleParameter 对象集合

ExecuteReader(); 执行查询 sql

ExecuteNonQuery 执行非查询 sql

✓ 连接

```
string connectstring=”Data Source=INSTANCE_NAME;user=xxx;password=xxxx;”;
```

✓ 实例化连接对象

```
OracleConnection conn = new OracleConnection(connectstr);
```

✓ 打开连接

```
Conn.Open();
```

✓ 定义操作 sql

OracleCommand cmd = conn.CreateCommand();

Cmd.CommandText = “select * from test”; // cmd.CommandType 可以指定操作类型如 sql 语句和存储过程）

✓ 执行 sql

OracleDataReader reader1 = cmd.ExecuteReader();

✓ 读取数据

```
While(reader1.Read())
{
}
```

✓ 取出数据

```
OracleString str = reader1.GetOracleString(0); // 取出第一列 oracle 字符串
OracleNumber num = reader1.GetOracleNumber(1); // 取出第二列， oracle 中的 number 数据类型
OracleDateTime = reader1.GetOracleDateTime(2) ; //取出第三； 时间类型
OracleBinary bin = reader1.GetOracleBinary(3) ; // 取出第四列,二进制数据
If (bin.IsNull == false)
{
    Foreach (byte b in bin.Value)
    {
        ….
    }
}
```

✓ 关闭连接

```
Conn.Close();
```

✓ 存储过程开发：

Oralce 存储过程特点：

Oracle 存储过程不支持 return 参数（只在函数中使用），只支持 out 参数返回结果

如果需要返回结果集，只能使用 REF CURSOR 参数

代码：

```
//定义 command
cmd.CommandText=”storename”;
cmd.CommandType=StoredProcedure;
//定义输出参数
cmd.Parameters.Add(“field”,OracleType.Number); //如果是数据集，则需要定义为游标类型
cmd.Parameters[“field”].Direction=ParameterDirection.Output;
//定义传入参数
cmd.Parameters.Add(“fiedl”,OracleType.xxx).Value = xxx;
// 执行
Conn.open();
OracleDataReader dr=cmd.ExecuteReader();// 如果没有结果集返回则使用函数：ExecuteNonQuery
//获取返回值
```

```
While(dr.Read())
{
    For(int I = 0 ; I < dr.FieldCount; I ++)
        ….
}
//非结果集的 out 参数提取
Cmd.Parameters[“field”].Value;
```

注意： 访问 ORACLE 同 sqlserver 有个不同的地方， 我们需要在 ORALCE 安装目录的 NETWORK/ADMIN 下的 tnsnames.ora， 该文件可以通过 oracle 自带工具配置

② OLDB

1、使用 OleDbConnection 类

2、连接字符串：ConnectionString=Provider=OraOLEDB.1;Password=password;User ID=username;Data Source=kdc(INSTANCE_NAME)

③ 结论

1、最好使用 oralceclient 连接数据库， 该命名空间对 ORACLE 连接管理进行了优化，并且支持所有 oralce 数据类型；
2、使用 OLEDB 的缺点有：无法调用 oracle 存储过程，只能执行 sql 语句，原因：oracle 数据库中的存储过程返回值是通过游标的方式返回的，不象其他数据库通过数据集、表的方式)

6. ORACLE 数据库的恢复与备份

① 备份种类

- ✓ 物理备份

  将实际组成数据库的操作系统文件从一处拷贝到另一处的备份过程，涉及：数据文件、控制文件、联机日志文件、ini 文件、其他配置文件。物理备份包括

  - ✧ 冷备份

    数据库在关闭的状态下进行备份

  - ✧ 热备份

    数据库在打开的状态下进行备份，且只能对数据文件和控制文件进行备份，热备份时数据库必须运行于归档模式。

- ✓ 逻辑备份

  利用 SQL 语言从数据库中抽取数据并存于二进制文件的过程。逻辑备份可按数据库中某个表、某个用户或整个数据库来导出，并且支持全部、累计、增量三种方式。使用这种方法，数据库必须处于打开状态，而且如果数据库是在 restrict 状态将不能保证导出数据的一致性。

② 恢复的种类

- ✓ 实例恢复
  Oracle 实例出现失败后，Oracle 自动进行的恢复
- ✓ 介质恢复
  数据库的介质出现故障时所做的恢复，可以分为：
  - ✧ 完全恢复
    将数据库恢复到数据库失败时数据库的状态。这种恢复是通过装载数据库备份并应用全部的重做日志做到的
  - ✧ 不完全恢复
    将数据库恢复到数据库失败前的某一时刻数据库的状态。这种恢复是通过装载数据库备份并应用部分的重做日志做到的，进行不完全恢复后须在启动数据库时用 resetlogs 选项重设联机重做日志（做增量备份时，用户必须具有 EXP FULL DATABASE 角色）

7. Sys 密码修改办法

① Unix 下

清空 oracle 的 sys 密码文件，位于 $ORACLE_HOME/dbs/orapw<sid>
创建密码文件： orapw file=$ORACLE_HOME/dbs/orapw<sid> password=????

② Windows

清空$ORACLE_HOME/database\pwd<sid>.ora
执行： 如 unix 命令

8. 命令

① 更改日期格式

```
Alter session set nls_date_format 'YYYY-MM-DD'
Alter session set nls_date_language='SIMPLIFIED CHINESE'
```

① 显示服务器配置文件

```
Show parameter spfile
Create spfile from file
```

① 修改表

a. 给表增加一列

alter table table_name add (col1 type,col2 type);

b. 修改列名

alter table table_name rename old_name to new_name

c. 删除字段

alter table table_name drop field_name

d. 修改列属性

alter table table_name modify(field_name new_attr)

e. 设置列无用状态

alter table table_name set unused column field_name

f. 删除无用字段

alter table table_name drop unused colomns

g. 增加主键

alter table table_name add constrain key_name primary key(field_name_list);

h. 修改字段允许为空

alter table conf_symbolstyle modify fieldcondition null

i. 修改字段名

alter table conf_objectmeta rename column othertypeid2 to othertypeid;

① 修改表空间数据文件

a. 增加数据文件

Alter tablespace tablespacename
Add datafile 'xxxxxxxxxxxxpath/name' size 500m
Autoextend on next 50m
Maxsize 1000m

a. 修改数据文件属性

Alter database datafile
'xxxxxxxxxxxxxxxxxxxxxpath/filename'
Autoextend on
Next 100m maxsize 1000m

a. 创建表空间

Create tablespace sales
Datafile 'xxxxxxxxxxxxxxxxxxxxxpath/filename1' size 500m
Autoextend on
Next 50m
Maxsize 1000m,
'yyyyyyyyyyyyyyyyyyyyyypath/filename2' size 500m
Autoextend off,

'zzzzzzzzzzzzzzzzzzzzzzzzzzzzypath/filename2' size 500m

Autoextend on

Next 50m maxsize unlimited;

a. 更改数据文件大小（扩大、压缩）

Alter database datafile

'xxxxxxxxxxxxxxxxxxxxpath/filename' resize 800m

select * from A ORDER BY OID desc

a. 修改表空间为只读

Alter tablespace temp online;

Alter tablespace users read write;

Alter tablespace users offlne;

Alter tablespace users read only;

系统表空间 system 不能设置 offline 和 read only

① 数据库创建

Create database geoSg

Logfile '/u01/app/oracle/oradata/geoSg/redo01.log'

size 10m,

Logfile '/u01/app/oracle/oradata/geoSg/redo02.log'

size 10m,

Logfile '/u01/app/oracle/oradata/geoSg/redo03.log'

size 10m,

Maxlogfiles 100

Maxlogmembers 3

Maxloghistory 226

Maxinstances 8

Maxdatafiles 100

Datafile '/u01/app/oracle/oradata/geoSg/geosg.dbf' size 400m reuse

Autoextend on next 640k maxsize unlimited

Character set zhs16gbk

National charachter set zhs16gbk

注意数据库创建，必须将数据库启动到 nomount 状态

9. 视图相关操作

① 查询方案具有的视图

select view_name from dba_views where lower(owner)='geostar';

10. 数据库资源解锁

① 查看锁

Select t2.username,t2.sid,t2.serial#,t2.logon_time from v$locked_object t1,v$session t2 where t1.session_id=t2.sid order by t2.logon_time;

② 解除锁

alter system kill session 'sid,serial#';
alter system kill session'14,3693';

11. 日志文件相关

日志文件管理，操作员必须具有 ALTER DATABASE 权限。

① 日志文件查看

Select * from v$logfile;

② 查看日志是否归档

Select * from v$log;

③ 日志文件切换

Alter system switch logfile;

④ 查看数据库归档模式

Select dbid, name ,created,log_mode from v$database;
或 SQL> archive log list

⑤ 手动归档日志文件

Alter system archive log all

⑥ 日志文件影像

日志文件影像，则是通过日志文件分布在不同磁盘，来保证日志文件的可用性
可以在创建日志组时，指定日志文件分布不同的磁盘
已经存在的日志文件，可以通过添加日志文件的方式存放到不同的磁盘

⑦ 增加日志文件

--增加新组
Alter database add logfile
('/u01/app/oracle/oracle/oradata/orcl/log4a.log',
'/u01/app/oracle/oracle/oradata/orcl/log4b.log',
'/u01/app/oracle/oracle/oradata/orcl/log4c.log') size 100m;

-- 在已知日志组中添加成员
Alter database add logfile member
'/u01/app/oracle/oracle/oradata/orcl/log1b.log' to group 1;

添加的日志文件大小为当初创建日志组的大小一致，一个日志组里头的日志文件大小相同。

⑧ 添加日志文件组

Alter database add logfile group 4 (‘/path/file1.log’,’/path/file2.log’) size 4m;

⑨ 删除日志组

Alter database drop logfile group id //id 表示组号， 通过 v$logfile 可以查询
当前激活日志组不可以删除，如果需要删除，请切换日志。
该命令没有删除日志物理文件，如果需要删除，则需要通过操作系统命令清除。

⑩ 删除日志文件

Alter database drop logfile member '/u01/app/oracle/oracle/oradata/orcl/log1b.log' ;
正在使用、未归档的日志文件不允许删除。

11 移动日志文件

Step1：connect / as sysdba
Step2：shutdown immediate
Step3：mv logfile
Step4：Alter database rename file
'/u01/app/oracle/oracle/oradata/orcl/log1b.log' to

'/u02/app/oracle/oracle/oradata/orcl/log1b.log' ;

12 清空（初始化）日志

SQL> Alter database clear unarchived logfile group 1;
SQL>recover database until cancel;
SQL> alter database open resetlogs;

如果联机日志文件发生冲突，则清空日志文件组
Alter database clear logfile group 5;
如果待初始化的日志未归当，则用下面的命令
SQL> Alter database clear unarchived logfile group 1;
如果待清空的日志文件，表空间重新在线，则必须删除表空间
Alter database clear logfile group 3 unrecoverable datafile;

WARNING :联机日志文件初始化，危险性太大，请谨慎操作。

13 日志文件状态

Current ： 表示 LWGR 正在使用的日志文件，正在写入
Active： 表示 LWGR 正在写入的日志文件，实例恢复时将需要使用
Inactive： 表示 LWGR 正在写入的日志文件，实例恢复时不使用

14 更改日志切换时间

Alter system set archive_lag_target=60 scope=both;
日志切换时间以秒为单位，时长范围为 60~7200 秒，如果 archive_lag_target=0 则表示不使用基于时间的日志切换。
查看日志文件大小
Select group#,bytes/1024/1024||'M' from v$log

15 更改日志文件大小

Step1：查询当前日志文件组大小(v$log)
Step2：创建临时日志文件组
Step3：使当前日志文件组归档
Step4：切换归档日志到临组
Step5：删除待修改的日志文件组
Step6：使用操作系统命令，删除日志文件物理文件(v$log 记录了位置)
Stpe7：重建日志文件组，并指定新的大小
Step8：使临时日志文件组归档
Step9：切换日志文件组到更改后的日志文件组

16 错误处理

- Question1:

现象：创建日志文件组时提示：ora-00336:log file size 4096 blocks is less than minimum 8129 blocks

原因：oracle 对日志文件最小有限制，指定日志文件过小导致的

解决：更改日志文件大小

- Question2：

现象：删除日志组，提示：ora-01642: log 2 needed for crash recovery of instance nbo(thread 1)

原因：日志文件组没有归档， 状态为 active

解决：

查看状态

Select group#,bytes/1024/104||'M',status ,archived from v$log;

手动产生 checkpoint

Alter system checkpoint

再查看日志组状态

Select group#,bytes/1024/104||'M',status ,archived from v$log;

再删除日志组

Alter database drop logfile group 2

- Question3:

现象：删除日志文件组时提示：ora-00350 log 3 of thread 1 needs to be archived

原因：试图删除未归档的日志文件组，如果数据运行在归档模式，这是不允许的

解决：

alter system archive log all

alter database drop logfile group 3

- question4：

现象：删除日志文件组提示：ora-01623:log 2 is current log for instance nbo (thread1)-cannot drop

原因：不能删除，状态为 current 的日志文件组

解决：切换日志（alter system switch logfile）

- Question5：

现象:添加联机日志文件时,提示:ora-00357: too many members specified for log file , the maximum is 3

原因：达到日志组文件上限，缺省为 3， 需要重建控制文件，吧 MAXLOGMEMBERS 改大。

12. 控制文件备份步骤

① 关闭数据库

a) -- shutdown

② 复制控制文件 -

b) --copy / cp

③ 修改参数文件

c) ---- 修改控制文件路径和名称 init.ora

④ 重启数据库

d) ----- startup

13. ORACLE 常用函数

① 字符函数

- ASCII

返回与指定的字符对应的十进制数;

SQL> select ascii(’A’) A,ascii(’a’) a,ascii(’0’) zero,ascii(’ ’) space from dual;

A A ZERO SPACE

--------- --------- --------- ---------

65 97 48 32

- CHR

给出整数,返回对应的字符;

SQL> select chr(54740) zhao,chr(65) chr65 from dual;

ZH C

-- -

赵 A

- CONCAT

连接两个字符串;

SQL> select concat(’010-’,’88888888’)||’转 23’ 高乾竞电话 from dual;

高乾竞电话

010-88888888 转 23

- INITCAP

返回字符串并将字符串的第一个字母变为大写;

SQL> select initcap(’smith’) upp from dual;

UPP

Smith

- INSTR(C1,C2,I,J)

在一个字符串中搜索指定的字符,返回发现指定的字符的位置;

C1 被搜索的字符串

C2 希望搜索的字符串

I 搜索的开始位置,默认为 1

J 出现的位置,默认为 1

SQL> select instr(’oracle traning’,’ra’,1,2) instring from dual;

INSTRING

9

- LENGTH

返回字符串的长度;

SQL> select name,length(name),addr,length(addr),sal,length(to_char(sal)) from gao.nchar_tst;

NAME LENGTH(NAME) ADDR LENGTH(ADDR) SAL LENGTH(TO_CHAR(SAL))

------ ------------ ---------- ---------- --------- ----------

高乾竞 3 北京市海锭区 6 9999.99 7

- LOWER

返回字符串,并将所有的字符小写

SQL> select lower(’AaBbCcDd’)AaBbCcDd from dual;

AABBCCDD

---

aabbccdd

- UPPER

返回字符串,并将所有的字符大写

SQL> select upper(’AaBbCcDd’) upper from dual;

UPPER

AABBCCDD

- RPAD 和 LPAD(粘贴字符)

RPAD 在列的右边粘贴字符

LPAD 在列的左边粘贴字符

SQL> select lpad(rpad(’gao’,10,’*’),17,’*’)from dual;

LPAD(RPAD(’GAO’,1

*******gao*******

不够字符则用*来填满

- LTRIM 和 RTRIM

LTRIM 删除左边出现的字符串

RTRIM 删除右边出现的字符串

SQL> select ltrim(rtrim(’ gao qian jing ’,’ ’),’ ’) from dual;

LTRIM(RTRIM(’

---

gao qian jing

- BSTR(string,start,count)

取子字符串,从 start 开始,取 count 个

SQL> select substr(’13088888888’,3,8) from dual;

SUBSTR(’

08888888

- REPLACE(’string’,’s1’,’s2’)

string 希望被替换的字符或变量

s1 被替换的字符串

s2 要替换的字符串

SQL> select replace(’he love you’,’he’,’i’) from dual;

REPLACE(’H

----

i love you

- SOUNDEX

返回一个与给定的字符串读音相同的字符串

SQL> create table table1(xm varchar(8));

SQL> insert into table1 values(’weather’);

SQL> insert into table1 values(’wether’);

SQL> insert into table1 values(’gao’);

SQL> select xm from table1 where soundex(xm)=soundex(’weather’);

XM

----

weather

wether

- TRIM(’s’ from ’string’)

LEADING 剪掉前面的字符

TRAILING 剪掉后面的字符

如果不指定,默认为空格符

② 数学运算

- ABS

返回指定值的绝对值

SQL> select abs(100),abs(-100) from dual;

ABS(100) ABS(-100)

--------- ---------

100 100

- ACOS

给出反余弦的值

SQL> select acos(-1) from dual;

ACOS(-1)

3.1415927

- ASIN

给出反正弦的值

SQL> select asin(0.5) from dual;

ASIN(0.5)

.52359878

- ATAN

返回一个数字的反正切值

SQL> select atan(1) from dual;

ATAN(1)

.78539816

- CEIL

返回大于或等于给出数字的最小整数

SQL> select ceil(3.1415927) from dual;

CEIL(3.1415927)

4

- COS

返回一个给定数字的余弦

SQL> select cos(-3.1415927) from dual;

COS(-3.1415927)

-1

- COSH

返回一个数字反余弦值

SQL> select cosh(20) from dual;

COSH(20)

242582598

- EXP

返回一个数字 e 的 n 次方根

SQL> select exp(2),exp(1) from dual;

EXP(2) EXP(1)

--- ---

7.3890561 2.7182818

- FLOOR

对给定的数字取整数

```
SQL> select floor(2345.67) from dual;
FLOOR(2345.67)
--------------
2345
```

- LN

返回一个数字的对数值

```
SQL> select ln(1),ln(2),ln(2.7182818) from dual;
LN(1) LN(2) LN(2.7182818)
--------- --------- -------------
0 .69314718 .99999999
```

- LOG(n1,n2)

返回一个以 n1 为底 n2 的对数

```
SQL> select log(2,1),log(2,4) from dual;
LOG(2,1) LOG(2,4)
--------- ---------
0 2
```

- MOD(n1,n2)

返回一个 n1 除以 n2 的余数

```
SQL> select mod(10,3),mod(3,3),mod(2,3) from dual;
MOD(10,3) MOD(3,3) MOD(2,3)
--------- --------- ---------
1 0 2
```

- POWER

返回 n1 的 n2 次方根

```
SQL> select power(2,10),power(3,3) from dual;
POWER(2,10) POWER(3,3)
----------- ----------
1024 27
```

- ROUND 和 TRUNC

按照指定的精度进行舍入

```
SQL> select round(55.5),round(-55.4),trunc(55.5),trunc(-55.5) from dual;
ROUND(55.5) ROUND(-55.4) TRUNC(55.5) TRUNC(-55.5)
----------- ------------ ----------- ------------
56 -55 55 -55
```

- SIGN

取数字 n 的符号,大于 0 返回 1,小于 0 返回-1,等于 0 返回 0

```
SQL> select sign(123),sign(-100),sign(0) from dual;
SIGN(123) SIGN(-100) SIGN(0)
--------- ---------- ---------
1 -1 0
```

- SIN

返回一个数字的正弦值

```
SQL> select sin(1.57079) from dual;
SIN(1.57079)
------------
1
```

- SIGH

返回双曲正弦的值

```
SQL> select sin(20),sinh(20) from dual;
SIN(20) SINH(20)
--------- ---------
.91294525 242582598
```

- SQRT

返回数字 n 的根

```
SQL> select sqrt(64),sqrt(10) from dual;
SQRT(64) SQRT(10)
--------- ---------
8 3.1622777
```

- TAN

返回数字的正切值

```
SQL> select tan(20),tan(10) from dual;
TAN(20) TAN(10)
--------- ---------
2.2371609 .64836083
```

- TANH

返回数字 n 的双曲正切值

```
SQL> select tanh(20),tan(20) from dual;
TANH(20) TAN(20)
--------- ---------
1 2.2371609
```

- TRUNC

按照指定的精度截取一个数

SQL> select trunc(124.1666,-2) trunc1,trunc(124.16666,2) from dual;

TRUNC1 TRUNC(124.16666,2)

--------- ----------

100 124.16

③ 时间日期

- ADD_MONTHS

增加或减去月份

SQL> select to_char(add_months(to_date(’199912’,’yyyymm’),2),’yyyymm’) from dual;

TO_CHA

200002

SQL> select to_char(add_months(to_date(’199912’,’yyyymm’),-2),’yyyymm’) from dual;

TO_CHA

199910

- LAST_DAY

返回日期的最后一天

SQL> select to_char(sysdate,’yyyy.mm.dd’),to_char((sysdate)+1,’yyyy.mm.dd’) from dual;

TO_CHAR(SY TO_CHAR((S

---------- ----------

2004.05.09 2004.05.10

SQL> select last_day(sysdate) from dual;

LAST_DAY(S

31-5 月 -04

- MONTHS_BETWEEN(date2,date1)

给出 date2-date1 的月份

SQL> select months_between(’19-12 月-1999’,’19-3 月-1999’) mon_between from dual;

MON_BETWEEN

9

SQL>selectmonths_between(to_date(’2000.05.20’,’yyyy.mm.dd’),to_date(’2005.05.20’,’yyyy.mm.dd’)) mon_betw from dual;

MON_BETW

-60

- NEW_TIME(date,’this’,’that’)

给出在 this 时区=other 时区的日期和时间

SQL> select to_char(sysdate,’yyyy.mm.dd hh24:mi:ss’) bj_time,to_char(new_time
2 (sysdate,’PDT’,’GMT’),’yyyy.mm.dd hh24:mi:ss’) los_angles from dual;
BJ_TIME LOS_ANGLES
------------------- -------------------
2004.05.09 11:05:32 2004.05.09 18:05:32

- NEXT_DAY(date,’day’)

给出日期 date 和星期 x 之后计算下一个星期的日期
SQL> select next_day(’18-5 月-2001’,’星期五’) next_day from dual;
NEXT_DAY

25-5 月 -01

- SYSDATE

用来得到系统的当前日期
SQL> select to_char(sysdate,’dd-mm-yyyy day’) from dual;
TO_CHAR(SYSDATE,’

09-05-2004 星期日
trunc(date,fmt)按照给出的要求将日期截断,如果 fmt=’mi’表示保留分,截断秒
SQL> select to_char(trunc(sysdate,’hh’),’yyyy.mm.dd hh24:mi:ss’) hh,
2 to_char(trunc(sysdate,’mi’),’yyyy.mm.dd hh24:mi:ss’) hhmm from dual;
HH HHMM
------------------- -------------------
2004.05.09 11:00:00 2004.05.09 11:17:00

④ 其他辅助

- CHARTOROWID

将字符数据类型转换为 ROWID 类型
SQL> select rowid,rowidtochar(rowid),ename from scott.emp;
ROWID ROWIDTOCHAR(ROWID) ENAME
------------------ ------------------ ----------
AAAAfKAACAAAAEqAAA AAAAfKAACAAAAEqAAA SMITH
AAAAfKAACAAAAEqAAB AAAAfKAACAAAAEqAAB ALLEN
AAAAfKAACAAAAEqAAC AAAAfKAACAAAAEqAAC WARD
AAAAfKAACAAAAEqAAD AAAAfKAACAAAAEqAAD JONES

- CONVERT(c,dset,sset)

将源字符串 sset 从一个语言字符集转换到另一个目的 dset 字符集
SQL> select convert(’strutz’,’we8hp’,’f7dec’) "conversion" from dual;
conver
------
strutz

- HEXTORAW

将一个十六进制构成的字符串转换为二进制

- RAWTOHEXT

将一个二进制构成的字符串转换为十六进制

- ROWIDTOCHAR

将 ROWID 数据类型转换为字符类型

- TO_CHAR(date,’format’)

SQL> select to_char(sysdate,’yyyy/mm/dd hh24:mi:ss’) from dual;

TO_CHAR(SYSDATE,’YY

2004/05/09 21:14:41

- TO_DATE(string,’format’)

将字符串转化为 ORACLE 中的一个日期

- TO_MULTI_BYTE

将字符串中的单字节字符转化为多字节字符

SQL> select to_multi_byte(’高’) from dual;

TO

--

高

- TO_NUMBER

将给出的字符转换为数字

SQL> select to_number(’1999’) year from dual;

YEAR

1999

- BFILENAME(dir,file)

指定一个外部二进制文件

SQL>insert into file_tb1 values(bfilename(’lob_dir1’,’image1.gif’));

- CONVERT(’x’,’desc’,’source’)

将 x 字段或变量的源 source 转换为 desc

SQL> select sid,serial#,username,decode(command,

2 0,’none’,

3 2,’insert’,

4 3,

5 ’select’,

6 6,’update’,

```
7 7,’delete’,
8 8,’drop’,
9 ’other’) cmd from v$session where type!=’background’;
SID SERIAL# USERNAME CMD
--------- --------- ------------------------------ ------
1 1 none
2 1 none
3 1 none
4 1 none
5 1 none
6 1 none
7 1275 none
8 1275 none
9 20 GAO select
10 40 GAO none
```

- **DUMP(s,fmt,start,length)**

DUMP 函数以 fmt 指定的内部数字格式返回一个 VARCHAR2 类型的值

```
SQL> col global_name for a30
SQL> col dump_string for a50
SQL> set lin 200
SQL> select global_name,dump(global_name,1017,8,5) dump_string from global_name;
GLOBAL_NAME DUMP_STRING
------------------------------ --------------------------------------------------
ORACLE.WORLD Typ=1 Len=12 CharacterSet=ZHS16GBK: W,O,R,L,D
```

- **EMPTY_BLOB()和 EMPTY_CLOB()**

这两个函数都是用来对大数据类型字段进行初始化操作的函数

- **GREATEST**

返回一组表达式中的最大值,即比较字符的编码大小.

```
SQL> select greatest(’AA’,’AB’,’AC’) from dual;
GR
--
AC
SQL> select greatest(’啊’,’安’,’天’) from dual;
GR
--
天
```

- **LEAST**

返回一组表达式中的最小值

```
SQL> select least(’啊’,’安’,’天’) from dual;
LE
--
```

啊

- UID

返回标识当前用户的唯一整数

SQL> show user

USER 为"GAO"

SQL> select username,user_id from dba_users where user_id=uid;

USERNAME USER_ID

------------------------------ ---------

GAO 25

- USER

返回当前用户的名字

SQL> select user from dual;

USER

GAO

- USEREVN

返回当前用户环境的信息,opt 可以是:

ENTRYID,SESSIONID,TERMINAL,ISDBA,LABLE,LANGUAGE,CLIENT_INFO,LANG,VSIZE

ISDBA 查看当前用户是否是 DBA 如果是则返回 true

SQL> select userenv('isdba') from dual;

USEREN

FALSE

SQL> select userenv('isdba') from dual;

USEREN

TRUE

SESSION

返回会话标志

SQL> select userenv('sessionid') from dual;

USERENV('SESSIONID')

152

ENTRYID

返回会话人口标志

SQL> select userenv('entryid') from dual;

USERENV('ENTRYID')

0

INSTANCE

返回当前 INSTANCE 的标志

SQL> select userenv('instance') from dual;

USERENV(’INSTANCE’)

1

LANGUAGE

返回当前环境变量

SQL> select userenv(’language’) from dual;

USERENV(’LANGUAGE’)

SIMPLIFIED CHINESE_CHINA.ZHS16GBK

LANG

返回当前环境的语言的缩写

SQL> select userenv(’lang’) from dual;

USERENV(’LANG’)

ZHS

TERMINAL

返回用户的终端或机器的标志

SQL> select userenv(’terminal’) from dual;

USERENV(’TERMINA

GAO

VSIZE(X)

返回 X 的大小(字节)数

SQL> select vsize(user),user from dual;

VSIZE(USER) USER

---------- ------------------------------

6 SYSTEM

- AVG(DISTINCT|ALL)

all 表示对所有的值求平均值,distinct 只对不同的值求平均值

SQLWKS> create table table3(xm varchar(8),sal number(7,2));

语句已处理。

SQLWKS> insert into table3 values(’gao’,1111.11);

SQLWKS> insert into table3 values(’gao’,1111.11);

SQLWKS> insert into table3 values(’zhu’,5555.55);

SQLWKS> commit;

SQL> select avg(distinct sal) from gao.table3;

AVG(DISTINCTSAL)

3333.33

SQL> select avg(all sal) from gao.table3;

AVG(ALLSAL)

2592.59

- MAX(DISTINCT|ALL)

求最大值,ALL 表示对所有的值求最大值,DISTINCT 表示对不同的值求最大值,相同的只取一次

SQL> select max(distinct sal) from scott.emp;

MAX(DISTINCTSAL)

5000

- MIN(DISTINCT|ALL)

求最小值,ALL 表示对所有的值求最小值,DISTINCT 表示对不同的值求最小值,相同的只取一次

SQL> select min(all sal) from gao.table3;

MIN(ALLSAL)

1111.11

- STDDEV(distinct|all)

求标准差,ALL 表示对所有的值求标准差,DISTINCT 表示只对不同的值求标准差

SQL> select stddev(sal) from scott.emp;

STDDEV(SAL)

1182.5032

SQL> select stddev(distinct sal) from scott.emp;

STDDEV(DISTINCTSAL)

1229.951

- VARIANCE(DISTINCT|ALL)

求协方差

SQL> select variance(sal) from scott.emp;

VARIANCE(SAL)

1398313.9

- GROUP BY

主要用来对一组数进行统计

SQL> select deptno,count(*),sum(sal) from scott.emp group by deptno;

DEPTNO COUNT(*) SUM(SAL)

--- --- ---

10 3 8750

20 5 10875

30 6 9400

- HAVING

对分组统计再加限制条件

SQL> select deptno,count(*),sum(sal) from scott.emp group by deptno having count(*)>=5;

DEPTNO COUNT(*) SUM(SAL)

--------- --------- ---------

20 5 10875

30 6 9400

SQL> select deptno,count(*),sum(sal) from scott.emp having count(*)>=5 group by deptno ;

DEPTNO COUNT(*) SUM(SAL)

--------- --------- ---------

20 5 10875

30 6 9400

- ORDER BY

用于对查询到的结果进行排序输出

SQL> select deptno,ename,sal from scott.emp order by deptno,sal desc;

DEPTNO ENAME SAL

--------- ---------- ---------

10 KING 5000

10 CLARK 2450

10 MILLER 1300

20 SCOTT 3000

20 FORD 3000

20 JONES 2975

20 ADAMS 1100

20 SMITH 800

30 BLAKE 2850

30 ALLEN 1600

30 TURNER 1500

30 WARD 1250

30 MARTIN 1250

30 JAMES 950

select value from nls_database_parameters;

14. 常用名词

① 服务名

数据库用服务名进行标识。由初始化参数文件中的 SERVICE_NAMES 参数指定。

② 实例名

数据库实例由 INSTANCE_NAME 参数的实例名标识，该标识对应实例的 SID

③ 连接描述符

配置连接字符串，包含数据库连接端口、连接的数据服务名、实例名（可选）、主机名、协议

④ 数据库的几种打开关闭方式说明

- ✓ Shutdown
  - ✧ normal:等待所有用户断开连接时，关闭数据库、卸载数据库和关闭实例。
  - ✧ immediate：回滚所有用户事务，关闭数据库、卸载数据库和关闭实例.（注意是回滚）。
  - ✧ transactional：当所有用户事务结束时，关闭数据库、卸载数据库和关闭实例
  - ✧ abort：立即终止实例。对用户未交事务，下次启动数据时恢复。

- ✓ statup
  - ✧ startup nomount

    非加载启动实例，如果在此状态下打开数据库需要的操作：

    Alter database mount

    Alter database open

    说明：nomount 是非安装启动，做的操作主要是读取 init.ora 文件，启动 instance，即启动 SGA 和后台进程，划分内存，进程。这种启动只需要读 init.ora 文件。这种启动方式下可执行：重建控制文件、重建数据库，因为还没有读控制文件。

  - ✧ startup mount

    启动实例>;装载数据库，如果在此状态下打开数据库需要的操作：

    Alter database open

    说明：mount 是安装启动，做的操作主要是：打开控制文件，确认数据文件和联机日志文件的位置，但此时不对数据文件和日志文件进行一致性，正确性等校验检查。这种启动下可执行： 数据库日志归档、数据库介质恢复、使数据文件联机或脱机、重新定位数据文件、重做日志文件。

  - ✧ statup open/startup 启动实例>;装载数据库

    说明：打开数据库这种方式又执行了打开包括 Redo log 文件在内的所有数据库文件，并检查一致性等，进行 crash 恢复，这种方式下可访问数据库中的数据。
  - ✧ statup force 强制重启数据库
  - ✧ 使用非缺省参数文件起动

    STARTUP PFILE=参数文件

    注：远程启动数据时，系统寻找本地计算机的参数文件。

  - ✧ STARTUP RESTRICT

    说明：通过以下命令改变当前数据库状态 ALTER SYSTEM [ENABLE|DISABLE] RESTRICTED SESSION 只有 RESTRICTED SESSION 权限的用户才能登录。

  - ✧ 只读方式打开数据

ALTER DATABASE OPEN READ ONLY

15. Oracle 字符集

Oracle 字符集是一个字节数据的解释的符号集合,有大小之分,有相互的包容关系。ORACLE 支持国家语言的体系结构允许你使用本地化语言来存储，处理，检索数据。它使数据库工具，错误消息，排序次序，日期，时间，货币，数字，和日历自动适应本地化语言和平台。

影响 Oracle 数据库字符集最重要的参数是 NLS_LANG 参数。

它的格式如下: NLS_LANG = language_territory.charset

它有三个组成部分(语言、地域和字符集)，每个成分控制了 NLS 子集的特性。

其中:

Language： 指定服务器消息的语言， 影响提示信息是中文还是英文

Territory： 指定服务器的日期和数字格式，

Charset： 指定字符集。

如:AMERICAN _ AMERICA. ZHS16GBK

从 NLS_LANG 的组成我们可以看出，真正影响数据库字符集的其实是第三部分。

所以两个数据库之间的字符集只要第三部分一样就可以相互导入导出数据，前面影响的只是提示信息是中文还是英文。

① 字符集的相关知识：

1) 字符集

实质就是按照一定的字符编码方案，对一组特定的符号，分别赋予不同数值编码的集合。Oracle 数据库最早支持的编码方案是 US7ASCII。

Oracle 的字符集命名遵循以下命名规则:

<Language><bit size><encoding>

即: <语言><比特位数><编码>

比如: ZHS16GBK 表示采用 GBK 编码格式、16 位（两个字节）简体中文字符集

2) 字符编码方案

a. 单字节编码

（1）单字节 7 位字符集，可以定义 128 个字符，最常用的字符集为 US7ASCII

（2）单字节 8 位字符集，可以定义 256 个字符，适合于欧洲大部分国家

例如：WE8ISO8859P1(西欧、8 位、ISO 标准 8859P1 编码)

b. 多字节编码

（1）变长多字节编码

某些字符用一个字节表示，其它字符用两个或多个字符表示，变长多字节编码常用于对亚洲语言的支持， 例如日语、汉语、印地语等

例如：AL32UTF8（其中 AL 代表 ALL,指适用于所有语言）、zhs16cgb231280

（2）定长多字节编码

每一个字符都使用固定长度字节的编码方案，目前 oracle 唯一支持的定长多字节编码是 AF16UTF16，也是仅用于国家字符集

c. unicode 编码

Unicode 是一个涵盖了目前全世界使用的所有已知字符的单一编码方案，也就是说 Unicode 为每一个字符提供唯一的编码。UTF-16 是 unicode 的 16 位编码方式，是一种定长多字节编码，用 2 个字节表示一个 unicode 字符，AF16UTF16 是 UTF-16 编码字符集。

UTF-8 是 unicode 的 8 位编码方式，是一种变长多字节编码，这种编码可以用 1、2、3 个字节表示一个 unicode 字符，AL32UTF8，UTF8、UTFE 是 UTF-8 编码字符集

3）字符集超级

当一种字符集（字符集 A）的编码数值包含所有另一种字符集（字符集 B）的编码数值，并且两种字符集相同编码数值代表相同的字符时，则字符集 A 是字符集 B 的超级，或称字符集 B 是字符集 A 的子集。

Oracle8i 和 oracle9i 官方文档资料中备有子集-超级对照表（subset-superset pairs），例如：WE8ISO8859P1 是 WE8MSWIN1252 的子集。由于 US7ASCII 是最早的 Oracle 数据库编码格式，因此有许多字符集是 US7ASCII 的超集，例如 WE8ISO8859P1、ZHS16CGB231280、ZHS16GBK 都是 US7ASCII 的超集。

4）数据库字符集（oracle 服务器端字符集）

数据库字符集在创建数据库时指定，在创建后通常不能更改。在创建数据库时，可以指定字符集(CHARACTER SET)和国家字符集(NATIONAL CHARACTER SET)。

5）字符集

(1)用来存储 CHAR, VARCHAR2, CLOB, LONG 等类型数据

(2)用来标示诸如表名、列名以及 PL/SQL 变量等

(3)用来存储 SQL 和 PL/SQL 程序单元等

6）国家字符集：

(1)用以存储 NCHAR, NVARCHAR2, NCLOB 等类型数据

(2)国家字符集实质上是为 oracle 选择的附加字符集，主要作用是为了增强 oracle 的字符处理能力，因为 NCHAR 数据类型可以提供对亚洲使用定长多字节编码的支持，而数据库字符集则不能。国家字符集在 oracle9i 中进行了重新定义，只能在 unicode 编码中的 AF16UTF16 和 UTF8 中选择，默认值是 AF16UTF16

7）查询字符集参数

可以查询以下数据字典或视图查看字符集设置情况

nls_database_parameters、props$、v$nls_parameters

查询结果中 NLS_CHARACTERSET 表示字符集，NLS_NCHAR_CHARACTERSET 表示国家字符集

8）修改数据库字符集

按照上文所说，数据库字符集在创建后原则上不能更改。不过有 2 种方法可行。

1. 如果需要修改字符集，通常需要导出数据库数据，重建数据库，再导入数据库数据的方式来转换。

2. 通过 ALTER DATABASE CHARACTER SET 语句修改字符集，但创建数据库

后修改字符集是有限制的，只有新的字符集是当前字符集的超集时才能修改数据库字符集，例如 UTF8 是 US7ASCII 的超集，修改数据库字符集可使用 ALTER DATABASE CHARACTER SET UTF8。

9) 客户端字符集（NLS_LANG 参数）

2.5.1 客户端字符集含义

客户端字符集定义了客户端字符数据的编码方式，任何发自或发往客户端的字符数据均使用客户端定义的字符集编码,客户端可以看作是能与数据库直接连接的各种应用，例如 sqlplus,exp/imp 等。客户端字符集是通过设置 NLS_LANG 参数来设定的。

10) NLS_LANG 参数格式

NLS_LANG=<language>_<territory>.<client character set>

Language: 显示 oracle 消息,校验，日期命名

Territory：指定默认日期、数字、货币等格式

Client character set：指定客户端将使用的字符集

例如：NLS_LANG=AMERICAN_AMERICA.US7ASCII

AMERICAN 是语言，AMERICA 是地区，US7ASCII 是客户端字符集

11) 客户端字符集设置方法

1)UNIX 环境

$NLS_LANG=“simplified chinese”_china.zhs16gbk

$export NLS_LANG

编辑 oracle 用户的 profile 文件

2)Windows 环境

编辑注册表

Regedit.exe --》 HKEY_LOCAL_MACHINE --》 SOFTWARE --》 ORACLE-HOME

12) NLS 参数查询

Oracle 提供若干 NLS 参数定制数据库和用户机以适应本地格式，例如有 NLS_LANGUAGE,NLS_DATE_FORMAT,NLS_CALENDER 等，可以通过查询以下数据字典或 v$视图查看。

NLS_DATABASE_PARAMETERS:显示数据库当前 NLS 参数取值，包括数据库字符集取值

NLS_SESSION_PARAMETERS： 显示由 NLS_LANG 设置的参数，或经过 alter session 改变后的参数值（不包括由 NLS_LANG 设置的客户端字符集）

NLS_INSTANCE_PARAMETE： 显示由参数文件 init<SID>.ora 定义的参数

V$NLS_PARAMETERS：显示数据库当前 NLS 参数取值

13) 2.5.5 修改 NLS 参数

使用下列方法可以修改 NLS 参数

（1）修改实例启动时使用的初始化参数文件

（2）修改环境变量 NLS_LANG

（3）使用 ALTER SESSION 语句，在 oracle 会话中修改

（4）使用某些 SQL 函数

NLS 作用优先级别：Sql function > alter session > 环境变量或注册表 > 参数文件 > 数据库默认参数

② EXP/IMP 与 字符集

1) EXP/IMP

Export 和 Import 是一对读写 Oracle 数据的工具。Export 将 Oracle 数据库中的数据输出到操作系统文件中, Import 把这些文件中的数据读到 Oracle 数据库中，由于使用 exp/imp 进行数据迁移时，数据从源数据库到目标数据库的过程中有四个环节涉及到字符集，如果这四个环节的字符集不一致，将会发生字符集转换。

EXP

```
-------------  --------------------  --------------
|imp 导入文件|<-|环境变量 NLS_LANG|<-|数据库字符集|
 ------------   ------------------    -------------
```

IMP

```
-------------  --------------------  --------------
|imp 导入文件|->|环境变量 NLS_LANG|->|数据库字符集|
 ------------   ------------------    -------------
```

四个字符集是

（1）源数据库字符集

（2）Export 过程中用户会话字符集（通过 NLS_LANG 设定）

（3）Import 过程中用户会话字符集（通过 NLS_LANG 设定）

（4）目标数据库字符集

2) 导出的转换过程

在 Export 过程中，如果源数据库字符集与 Export 用户会话字符集不一致，会发生字符集转换，并在导出文件的头部几个字节中存储 Export 用户会话字符集的 ID 号。在这个转换过程中可能发生数据的丢失。

例:如果源数据库使用 ZHS16GBK，而 Export 用户会话字符集使用 US7ASCII，由于 ZHS16GBK 是 16 位字符集,而 US7ASCII 是 7 位字符集，这个转换过程中，中文字符在 US7ASCII 中不能够找到对等的字符，所以所有中文字符都会丢失而变成“?? ”形式，这样转换后生成的 Dmp 文件已经发生了数据丢失。

因此如果想正确导出源数据库数据，则 Export 过程中用户会话字符集应等于源数据库字符集或是源数据库字符集的超集

3) 导入的转换过程

（1）确定导出数据库字符集环境

通过读取导出文件头，可以获得导出文件的字符集设置

（2）确定导入 session 的字符集，即导入 Session 使用的 NLS_LANG 环境变量

（3）IMP 读取导出文件

读取导出文件字符集 ID，和导入进程的 NLS_LANG 进行比较

（4）如果导出文件字符集和导入 Session 字符集相同，那么在这一步骤内就不需要转换，如果不同，就需要把数据转换为导入 Session 使用的字符集。可以看出，导入数据到数据库过程中发生两次字符集转换

第一次:导入文件字符集与导入 Session 使用的字符集之间的转换，如果这个转换过程不能正确完成，Import 向目标数据库的导入过程也就不能完成。

第二次:导入 Session 字符集与数据库字符集之间的转换。

③ 查看数据库字符集

涉及三方面的字符集，

1. oracel server 端的字符集;

2. oracle client 端的字符集;

3. dmp 文件的字符集。

在做数据导入的时候，需要这三个字符集都一致才能正确导入。

4.1 查询 oracle server 端的字符集

有很多种方法可以查出 oracle server 端的字符集，比较直观的查询方法是以下这种:

SQL> select userenv('language') from dual;

USERENV('LANGUAGE')

SIMPLIFIED CHINESE_CHINA.ZHS16GBK

SQL>select userenv(‘language’) from dual;

AMERICAN _ AMERICA. ZHS16GBK

4.2 如何查询 dmp 文件的字符集

用 oracle 的 exp 工具导出的 dmp 文件也包含了字符集信息，dmp 文件的第 2 和第 3 个字节记录了 dmp 文件的字符集。如果 dmp 文件不大，比如只有几 M 或几十 M，可以用 UltraEdit 打开(16 进制方式)，看第 2 第 3 个字节的内容，如 0354，然后用以下 SQL 查出它对应的字符集:

SQL> select nls_charset_name(to_number('0354','xxxx')) from dual;

ZHS16GBK

如果 dmp 文件很大，比如有 2G 以上(这也是最常见的情况)，用文本编辑器打开很慢或者完全打不开，可以用以下命令(在 unix 主机上):

cat exp.dmp |od -x|head -1|awk '{print $2 $3}'|cut -c 3-6

然后用上述 SQL 也可以得到它对应的字符集。

4.3 查询 oracle client 端的字符集

在 windows 平台下，就是注册表里面相应 OracleHome 的 NLS_LANG。还可以在 dos 窗口里面自己设置，

比如: set nls_lang=AMERICAN_AMERICA.ZHS16GBK

这样就只影响这个窗口里面的环境变量。

在 unix 平台下，就是环境变量 NLS_LANG。

$echo $NLS_LANG

AMERICAN_AMERICA.ZHS16GBK

如果检查的结果发现 server 端与 client 端字符集不一致，请统一修改为同 server 端相同的字符集。

补充：

(1).数据库服务器字符集

select * from nls_database_parameters

来源于 props$，是表示数据库的字符集。

(2).客户端字符集环境

select * from nls_instance_parameters

其来源于 v$parameter，表示客户端的字符集的设置，可能是参数文件，环境变量或者是注册表

(3).会话字符集环境

select * from nls_session_parameters

来源于 v$nls_parameters，表示会话自己的设置，可能是会话的环境变量或者是 alter session 完成，如果会话没有特殊的设置，将与 nls_instance_parameters 一致。

(4).客户端的字符集要求与服务器一致，才能正确显示数据库的非 Ascii 字符。

如果多个设置存在的时候，NLS 作用优先级别：Sql function > alter session > 环境变量或注册表 > 参数文件 > 数据库默认参数

字符集要求一致，但是语言设置却可以不同，语言设置建议用英文。如字符集是 zhs16gbk，则 nls_lang 可以是 American_America.zhs16gbk。

④ 修改 oracle 的字符集

按照上文所说，数据库字符集在创建后原则上不能更改。因此，在设计和安装之初考虑使用哪一种字符集十分重要。对数据库 server 而言，错误的修改字符集将会导致很多不可测的后果，可能会严重影响数据库的正常运行，所以在修改之前一定要确认两种字符集是否存在子集和超集的关系。一般来说，除非万不得已，我们不建议修改 oracle 数据库 server 端的字符集。特别说明，我们最常用的两种字符集 ZHS16GBK 和 ZHS16CGB231280 之间不存在子集和超集关系，因此理论上讲这两种字符集之间的相互转换不受支持。

不过修改字符集有 2 种方法可行。

1. 通常需要导出数据库数据，重建数据库，再导入数据库数据的方式来转换。

2. 通过 ALTER DATABASE CHARACTER SET 语句修改字符集，但创建数据库后修改字符集是有限制的，只有新的字符集是当前字符集的超集时才能修改数据库字符集，例如 UTF8 是 US7ASCII 的超集，修改数据库字符集可使用 ALTER DATABASE CHARACTER SET UTF8。

5.1 修改 server 端字符集(不建议使用)

1. 关闭数据库

SQL>SHUTDOWN IMMEDIATE

2. 启动到 Mount

SQL>STARTUP MOUNT;

SQL>ALTER SYSTEM ENABLE RESTRICTED SESSION;

SQL>ALTER SYSTEM SET JOB_QUEUE_PROCESSES=0;

SQL>ALTER SYSTEM SET AQ_TM_PROCESSES=0;

SQL>ALTER DATABASE OPEN;
SQL>ALTER DATABASE CHARACTER SET ZHS16GBK;
SQL>ALTER DATABASE national CHARACTER SET ZHS16GBK;
SQL>SHUTDOWN IMMEDIATE;
SQL>STARTUP

注意：如果没有大对象，在使用过程中进行语言转换没有什么影响，（切记设定的字符集必须是 ORACLE 支持，不然不能 start） 按上面的做法就可以。

若出现‘ORA-12717: Cannot ALTER DATABASE NATIONAL CHARACTER SET when NCLOB data exists’ 这样的提示信息，

要解决这个问题有两种方法

1. 利用 INTERNAL_USE 关键字修改区域设置,
2. 利用 re-create,但是 re-create 有点复杂,所以请用 internal_use

SQL>SHUTDOWN IMMEDIATE;
SQL>STARTUP MOUNT EXCLUSIVE;
SQL>ALTER SYSTEM ENABLE RESTRICTED SESSION;
SQL>ALTER SYSTEM SET JOB_QUEUE_PROCESSES=0;
SQL>ALTER SYSTEM SET AQ_TM_PROCESSES=0;
SQL>ALTER DATABASE OPEN;
SQL>ALTER DATABASE NATIONAL CHARACTER SET INTERNAL_USE UTF8;
SQL>SHUTDOWN immediate;
SQL>startup;

如果按上面的做法做,National charset 的区域设置就没有问题

5.2 修改 dmp 文件字符集

上文说过，dmp 文件的第 2 第 3 字节记录了字符集信息，因此直接修改 dmp 文件的第 2 第 3 字节的内容就可以‘骗’过 oracle 的检查。这样做理论上也仅是从子集到超集可以修改，但很多情况下在没有子集和超集关系的情况下也可以修改，我们常用的一些字符集，如 US7ASCII，WE8ISO8859P1，ZHS16CGB231280，ZHS16GBK 基本都可以改。因为改的只是 dmp 文件，所以影响不大。

具体的修改方法比较多，最简单的就是直接用 UltraEdit 修改 dmp 文件的第 2 和第 3 个字节。

比如想将 dmp 文件的字符集改为 ZHS16GBK，可以用以下 SQL 查出该种字符集对应的 16 进制代码: SQL> select to_char(nls_charset_id('ZHS16GBK'), 'xxxx') from dual;

0354

然后将 dmp 文件的 2、3 字节修改为 0354 即可。

如果 dmp 文件很大，用 ue 无法打开，就需要用程序的方法了。

5.3 客户端字符集设置方法

1)UNIX 环境

$NLS_LANG=“simplified chinese”_china.zhs16gbk

$export NLS_LANG

编辑 oracle 用户的 profile 文件

2)Windows 环境

编辑注册表

Regedit.exe --- 》 HKEY_LOCAL_MACHINE --- 》 SOFTWARE --- 》 ORACLE-HOME

或者在窗口设置：

set nls_lang=AMERICAN_AMERICA.ZHS16GBK

16. 数据库审计

① 查看是否启动审计

查看密码允许重试的次数

```
select * from dba_profiles where resource_name like 'FAILED_LOGIN_ATTEMPTS%';
```

17. 数据库归档

① 查看归档路径

Show parameter log_archive_dest

或者 ：select dest_name , destination,status from v$archive_dest ;

② 设置归档路径

Alter system set log_archive_dest_n='LOCATION=/u01/log/g01' SCOPE=both

LOCATION 表示为指定为归档路径；SCOPE=both 表示马上生效。

注意： 如果未给数据库设置任何归档路径，则归档日志将被存放到闪回区。

③ 切换到归档模式

Shutdown immediate

Connect / as sysdba

Startup mount

Alter database archivelog

Alter database open

(注意：如果是 10g 之前的版本，则还需要配置参数文件中的 log_archive_start=true)

④ 切换到非归档模式

Shutdown immediate
Connect / as sysdba
Startup mount
Alter database noarchivelog
Alter database open

⑤ 查看数据库归档状态

Archive log list

⑥ 手动归档联机日志文件

Connect sys@sid as sysdba
Alter system archive log all

⑦ 手动归档当前日志文件

Connect sys@sid as sysdba
Alter system archive log current

⑧ 手动启\停动归档进程

Connect sys@sid as sysdba
Alter system archive log start
Alter system archive log stop

⑨ 查看启动的归档进程

Select * from v$bgprocess where paddr<>'00' and name like '%ARC%'

⑩ 如何定时删除过期日志文件

命令： Find log_path ! -mtime -2 -exec rm -f '{}' ';'
将该命令写入脚本，并将脚本加入定时器
Crontab -e
加入内容：

20 3 * * * /home/oracle/del_script.sh >> logfilepath 2>&1 (2>&1 表示无任执行成功与否都写入日志文件）

11 查看数据库处于哪个归档模式

Connect sys@sid as sysdba
Archive log list 或者 select name,log_mode from v$database

12 如何查看数据库产生了那些归档日志文件

Select name, sequence#,completion_time from v$archive_log order by sequence# desc;

13 如何查看归档进程信息

Select * from v$bgprocess where paddr<>'00' and name like '%ARC%';
或者 ps -ef|grep ora_arc

14 如何删除归档日志文件

Select count(*) from v$archived_log where deleted <> 'YES'
Su - oracle
Rman
Connect target system

a. 删除所有日志

RMAN>
Run{
Allocate channel t1 type disk;
Delete force noprompt archivelog all;
Release channel t1;
}

b. 按时间删除

RMAN>
{
allocate channel t1 type disk;
Delete force noprompt archivelog until time "to_date('2006-09-11 23:59:59','yyyy-mm-dd h24:mi:ss')";
Release channel t1;
}

15 设置归档日志文件大小

取决于联机日志文件大小

16 归档相关 FAQ

a. **问题 1**

现象：

普通用户登录数据库，提示：archive error connect internal only ,until freed

诊断：

检查归档路径状态：

Select destination ,status from v$archive_dest;

检查归档路径权限

Cd /u01/log

Ls -l

配置权限

Su - root

Chown -R oracle:dba g01

Chmod -R 775 g01

总结：

产生以上错误无非三个原因：a、目录 oracle 无写入权限 b、归档目录不存在 c、归档目录空间已满；

另外，如果未指定任何归档目录，则归档文件将被存放到数据库闪回区，如果闪回区 size 不够大，则也会提示以上错误。

a. **问题 2**

现象：

手动归档时提示错误： less destinations availabe than specified by LOG_ARCHIVE_MIN_SUCCEED_DEST"

原因：

查看归档路数量

Show parameter log_archive_min_succeed_dest

查看归档路径状态

Select destination ,status from v$archive_dest;

检查归档路径权限

Cd /u01/log

Ls -l

检查成功归档的路径个数

Select destination ,status from v$archive_dest;

如果只有一个路径成功，则可以修改系统参数，来减少成功归档个数的最小值(缺省两个） 或者通过修复错误的归档路径来解决

Alter system set log_archive_min_succeed_dest=1

问题 3：

现象

设置归档模式时，提示错误：“instance recovery required, cannot set archivelog mod'

诊断

数据库出现不一致，都会出现如上问题， 如数据库异常关闭，shutdown abort

解决办法：(先起，后关，再准备归档)

```
Connect sys@sid as sysdba
Startup
Shutdown immediate
Startup mount
Alter database archivelog
Alter database open
```

18. DML 语言

1. 在查询语句中增加回车键

```
Select ‘hello’||chr(10)||’I am caojiaju’ from dual;
```

2. 查询前五条记录

```
Select * from sales where rowed <=5
```

3. 查询系统日期

```
Select to_char(sysdate,’yyyy-mm-dd hh24:mi:ss’) from dual;
```

4. DECODE 用法

```
Select product_id,
DECODE(SALE,1000,’LOW’,
                2000,’AAAA’,
                  3000,’CCCC’,
                  ‘unknow’)
From sales;
```

5. CASE 的用法

```
Example 1:
Select name,
     Case credit when 100 then ‘low’
                 When 5000 then ‘high’
                 Else ‘medium’ end
From customer;

Example2
Select name, (case when score < 60 then ‘D’
```

```
When score>=60 and score>80 then ‘A’
 Else ‘NO SCORE’ end)
From score;
```

6. 子查询

```
Select *
From
( select * from students where name like ‘att%’)
Where id <10007
```

7. 查询分布式事务

```
Select * from dba_2pc_pending
```

8. 手动提交/回滚

```
Commit force ‘transaction_id’
Rollback force ‘transaction_id’
```

9. 合并表记录到另一张表

```
Insert into student(id,name)
(select id,name from student1 where rownum <=5
Union all
Select id,name from student2 where rownum< 10)
```

10. 如何插入日期类型

```
Insert into user(id, birthday) values(100,to_date(‘2000-01-09 9:42:17’ ,’yyyy-mm-dd hh24:mi:ss));
```

11. 如何根据表 A 更新表 B 的数据

Example 1:

```
  Update target
  Set major=(select source.major from source  where source.id=target.id)
  Where exist(select source.id from source where source.id=target.id)
```

Example 2:

```
Update target
Set major=(select source.major from source where source.id=target.id)
Where target.id in(select source.id from source from source)
```

Example 3:

```
Update target
Set major=(select source.majior from source where source.id=target.id)
Where target.id=(select source.id from source where source.id=tartget.id)
```

12. 如何删除表中重复记录

方法 1：

```
Delete from table_student
Where rowed not in (select min(rowed) from student group by id)   //( ID 重复)
```

方法 2：

```
Delete from student a
Where rowed!=(select max(rowed) from student b where a.id=b.id)
```

方法 3：

```
Create table tmp_student
As
(select distinct id,name from student);

Truncate table student;

Insert into student
Select * from tmp_student;
```

13. 如何查询两个表中不相同的记录

```
Select id,name
From source
Where id not in (select id from target)
Union （查询并集） union all （查询并集-不去除相同部分）
Select id,name
From target
Where id not in(select id from source)
```

14. 如何找出 A 表比 B 表多的记录

方法 1：

```
Select id,name from A
MINUS（查询两结果的补集）
Select id,name from B
```

方法 2：(以 ID 重复为标准)

```
Select Id, major
From A
WHERE not exists (select t.id from b where b.id=a.id);
```

方法 3：(以 ID 重复为标准)

```
Select id,major
From A
Where id not in (select id from B);
```

15. 如何找出两个表相同的记录

方法 1：

```
Select id ,name from A
Intersect（查询交集）
Select id,name from B;
```

方法 2：

```
Select id,major from A
Where id in (select id from B)
```

方法 3：

```
Select id ,major from A
Where exists(select id from B where a.id=b.id)
```

16. 创建数据库远程连接

```
CREATE PUBLIC DATABASE LINK hq.acme.com
USING 'sales';
```

17. 无条件 insert all

```
INSERT    ALL
    INTO sal_history VALUES(EMPID,HIREDATE,SAL)
    INTO mgr_history VALUES(EMPID,MGR,SAL)
    SELECT employee_id EMPID, hire_date HIREDATE,
           salary SAL, manager_id MGR
```

```
FROM    employees
WHERE employee_id > 200;
```

18. 有条件的 INSERT ALL

```
INSERT ALL
  WHEN SAL > 10000 THEN
    INTO sal_history VALUES(EMPID,HIREDATE,SAL)
  WHEN MGR > 200     THEN
    INTO mgr_history VALUES(EMPID,MGR,SAL)
    SELECT employee_id EMPID,hire_date HIREDATE,
           salary SAL, manager_id MGR
    FROM     employees
    WHERE    employee_id > 200;
```

19. 转义符查询

```
select * from t11 where name like '%\_%' escape '\';
```

20. 时间相关

1.10.1.1 显示时区 ‘US/Eastern’的时差

```
SELECT TZ_OFFSET('US/Eastern') FROM DUAL;
```

1.10.1.2 按照当前会话的时区显示当前会话的时间

```
ALTER SESSION SET NLS_DATE_FORMAT = 'DD-MON-YYYY HH24:MI:SS';
ALTER SESSION SET TIME_ZONE = '-5:0';
SELECT SESSIONTIMEZONE, CURRENT_DATE FROM DUAL;
```

1.10.1.3 显示数据库所在的时区

```
SELECT DBTIMEZONE FROM DUAL;
```

1.10.1.4 显示会话所在的时区

```
SELECT SESSIONTIMEZONE FROM DUAL;
```

1.10.1.5 从 HIRE_DATE 中抽出月

```
SELECT last_name, hire_date,
       EXTRACT (MONTH FROM HIRE_DATE)      //YEAR , DAY
FROM employees
WHERE manager_id = 100;
```

19. 备份与恢复概述

玩 Oracle 也有 2 年的时间了， 零零散散的也整理一些资料。 东西一多了，就理不清楚。 所以结合张晓明的《大话 Oracle RAC》的一些内容，和自己整理的一些笔记，对 Oracle 的备份和恢复做了一个系统的整理。 也是自己对知识的一个巩固吧。

1. 准备知识

先来看一些准备知识，了解 Oracle 的物理结构，有如下 4 种。

✓ 数据文件和数据块

Oracle 数据库的存储空间是用表空间来表示的，表空间只是一个逻辑概念，而物理上每个表空间是由磁盘文件组成，这些文件叫做数据文件（Data file），每个表空间可以由一个到多个数据文件组成，每个数据文件被划分为若干个最小的存储单位： 数据块（data block）。

Oracle 的用户数据是写到数据块上的，Oracle 是在 SGA 上操作数据的，修改数据或者添加数据都是在内存中进行，这些被修改的内存不会立即写入磁盘，而是以特定的时间间隔被写入磁盘。 如果哦数据库正常关闭，则在关闭之前将内存中的数据同步到磁盘，这时数据状态是一致的。如果数据库不正常关闭（如宕机，shutdown abort），内存中的内容没有完全写回磁盘，这时数据文件是不一致的。 如果数据文件是从备份中恢复出来的，数据文件也是不一致的，不一致性的数据文件必须恢复到一致的状态。

✓ 日志文件

Oracle 数据库日志文件包括 联机日志 和 归档日志， 这些文件都是用来记录数据库修改历史的。Oracle 数据库至少要有两组联机日志，联机日志循环使用，当一组联机日志写满后，就要切换到另一组联机日志，后者的内容就会被覆盖，这个过程叫作日志切换（Log Switch），在日志切换时会触发检查点（CheckPoint）。

数据库的修改操作要记录到日志文件中，并且这个记录动作是在修改数据之前进行的，正因为日志文件中记录了所有的修改历史，因此如果有过去某个时点的备份文件，并且有从那是到当前的所有日志文件，就可以通过在备份文件上“重演”这些日志的方式，把数据文件恢复到当前状态或者之间的任何时点的状态。

✓ 日志线程（Redo Thread）：

每个实例用到的联机日志就是一个 Redo Thread，单实例有且仅有一个 Redo Thread。在 RAC 环境下，每个实例都需要自己的联机日志，也就是每个实例都有自己的 Redo Thread。 这种每实例一个 Redo Thread 的设计就是为了避免实例间共享 Redo 文件引发的竞争，提高系统性能。 但是这也带了一个问题，就是用 RMAN 备份 RAC 的时候，需要所有的日志文件。 故需要在两个节点上互相把日志传送到另一个节点。

Thread 这个参数用来指定实例使用的 Redo Thread 线程号。一般和该实例的 INSTANCE_NUMBER 参数相同。通过视图 V$LOG 的 Thread#列可以确定日志组所属的线程。

注意： 因为 RAC 环境下有多个日志线程，所以在添加日志时必须指定线程号。

SQL>alter database add logfile thread 1 group 5 ('/oracle/oradata/redo5') size 50m;

在 RAC 环境下，用户操作是分布在多个实例之间的，各实例都有自己的联机日志，恢复时必须把所有实例的联机日志都合并，把 Redo Log Record 按照 SCN 排序，才能整理出准确的用户操作记录，所以 RAC 的联机日志必须放在共享存储上，以保证实例都能访问其他实例的联机日志。

在看一个 sql 的查询结果：

SQL> select * from v$log;

GROUP#	THREAD#	SEQUENCE#	BYTES	MEMBERS	ARC	STATUS
1	1	24585	104857600	1	YES	ACTIVE
2	1	24583	104857600	1	YES	INACTIVE
3	1	24584	104857600	1	YES	INACTIVE
8	1	24586	104857600	1	NO	CURRENT

从查询结果上，我们可以看到联机日志有 3 个状态。

INACTIVE: 表示 DBWR 已经做完，该日志包含的数据修改已经写到数据文件

ACTIVE: DBWR 没有做完，数据还没有写到数据文件。

CURRENT：当前正在使用的日志。 没有 DBWR 操作。

RedoLog Checkpoint 和 SCN 关系

http://blog.csdn.net/tianlesoftware/archive/2010/01/25/5251916.aspx

Redo Log 和 Checkpoint not complete

http://blog.csdn.net/tianlesoftware/archive/2009/12/01/4908066.aspx

Oracle 归档与非归档的切换

http://blog.csdn.net/tianlesoftware/archive/2009/10/19/4693470.aspx

log file sync(日志文件同步) 与 Log file parallel write 等待事件

http://blog.csdn.net/tianlesoftware/archive/2009/12/02/4916671.aspx

✓ 控制文件

控制文件记录了数据库的物理结构和状态（比如数据文件名，每个数据文件的检查点号，联机状态），包括备份和恢复的信息也记录在控制文件中。 恢复过程要根据控制文件中的信息，比如数据库的检查点，当前联机日志，数据文件检查点等来进行恢复操作，如果控制文件丢失，则恢复的过程会很艰难。

控制文件里包含的具体信息，参考我的 Blog：

Oracle 控制文件

http://blog.csdn.net/tianlesoftware/archive/2009/12/13/4974440.aspx

✓ Undo Segment

修改记录过程中，记录修改之前的状态会被记录到 Undo Segment 中， 这条记录叫作前镜像（before images）。 当需要撤销修改比如执行 Rollback 时，就用这条前镜像覆盖现有记录，对于 Insert 操作，前镜像就是一个空记录，对于 Undate，Delete 操作，前镜像就是修改之前的记录。

在数据库恢复过程中，日志和 Undo Segment 共同起作用，二者保证了最终恢复的一致状态。二者也对应了恢复

的两个阶段： 前滚（Roforware）和 回滚（Rollback）。在前滚阶段，在文件上重演日志内容，以把文件恢复到数据库关闭时的状态，但是数据库关闭时可能有很多修改操作没有提交，这些操作必须进行回滚，这就要利用 Undo Segment 的内容。

Oracle undo 回滚段管理

http://blog.csdn.net/tianlesoftware/archive/2009/11/30/4901666.aspx

2. 二．备份

备份是指数据的拷贝，这个拷贝可以用来重建数据库。 备份可以分为物理备份和逻辑备份。

物理备份： 指对数据文件，控制文件，联机日志文件等文件进行物理拷贝的方法，这种方法是在文件层进行的，通过冗余的文件备份来体统数据保护。物理备份又可分为联机备份（也叫作联机热备）和脱机备份（也称冷备）。

逻辑备份： 利用 Oracle 提供的导出工具把重要数据导出到文件，以后恢复时在利用工具把数据重新导入到数据库中，这种保护是在数据层进行的。 该类工具有： exp/imp, expdp/impdp（10g 以后版本）。

物理备份是最强健的数据保护方也，也是备份策略中首选的方法，逻辑备份只能作为物理备份的补充手段，不足以保护数据丢失。

物理备份又可分为用户管理备份（User-Managed Backup）和 RMAN 备份（Recovery Manager）。 前者是联合使用 SQL 命令和 OS 的 cp 命来进行文件备份。 Rman 备份指利用 RMAN 工具来进行备份。

Rman 备份有几点好处：

1. 增量备份
2. 数据块恢复：可以在新进行数据块恢复，不必进行数据文件恢复，提高系统的可用性。
3. 压缩备份
4. 加密备份

➢ RMAN 备份实例

a) 检查数据库模式：

sqlplus /nolog

conn /as sysdba

archive log list (查看数据库是否处于归档模式中)

若为非归档,则修改数据库归档模式。

startup mount

alter database archivelog

alter database open

b) 连接到 target 数据库

命令： connect target system @caojj123@SID

说明： connect target system/oracle@ora10g,如果数据库没有起来，也可要直接在 rman 命令下用 startup 进行启动数据库

c) 查看有无备份的东西

命令：list backupset 命令查看有没有备份的东西

d) 常用备份命令：

全库备份

RMAN> backup database plus archivelog delete input; 或 backup database tag='xxxname';

(备份全库及控制文件、服务器参数文件与所有归档的重做日志，并删除旧的归档日志)

表空间备份

RMAN> backup tablespace system plus archivelog delete input; (备份指定表空间及归档的重做日志，并删除旧的归档日志)_

备份归档日志

RMAN> backup archivelog all delete input;

RMAN 参数查看方法

show all ;

0 级增量备份

概念：全备份和 0 级增量备份。全备份和 0 级增量备份几乎是一样的。唯一的区别，0 级增量备份能作为增量备份的基础，而全备份不能作为增量备份的基础。其它方面完全一致

1.backup incremental level=0 database;(增量为 0 的备份)

2.backup incremental level 1 database;(增量为 1 的备份)

0 级增量备份，备份了 datafile,controlfile 和 parameter file

0 级增量备份没有备份的文件有：归档日志，重做日志和口令文件没有备份（口令文件不需要备份，我们用 orapw 来创建一个）

archivelog 在 nocatalog 模式下的备份

全库备份：

命令:backup database plus archivelog delete input(delete input 的意思在备份完成后，删除 archivelog 文件，这个选项可要可不要，这个命令也可以用 backup incremental level=0(1,2...)来进行备份)

备份表空间

backup tablespace tablespacename

如果我们不知道 tablespace 的名字，在 rman 中，可要通过 report schema 命令，来查看表空间的名字

查看表空间信息

MAN> report schema;

备份控制文件

backup current controlfile

backup database include current controlfile

备份镜像

在 rman 的备份中有两种方式:备份集(backupset)和备份镜像(image copies).镜像备份主要是文件的

拷贝:copy datafile ... to ...

rman>copy datafile 5 to '/u01/rmanbak/tbso1bak.dbf';(copy 5 对应的 schme:perfstat.dbf)

它会把 tbs 作为一个拷贝。我们用 list backupset 来看，不能够查看我们刚备份的 tbs01bak.dbf', 因为它不是 backupset. 我们用 list copy 就能够查看我们刚才刚刚备份的文件

==========单命令与批命令==========

单命令: backup database;

批命令:

```
rman> run{
 2> allocate channel cha1 type disk;
 3> backup
 4> format '/u01/rmanbak/full_%t'
 5> tag full-backup //标签可以顺便起，没关系
 6> database;
 7> release channel cha1;
 8>}
```

这个 run 中有 3 条命令，分别用分号来进行分割.

format:

%c：备份片的拷贝数(从 1 开始编号);

%d：数据库名称;

%D：位于该月中的天数(DD);

%M：位于该年中的月份(MM);

%F：一个基于 DBID 唯一的名称，这个格式的形式为 c-xxx-YYYYMMDD-QQ,其中 xxx 位该数据库的 DBID，YYYYMMDD 为日期，QQ 是一个 1-256 的序列;

%n：数据库名称，并且会在右侧用 x 字符进行填充，使其保持长度为 8;

%u:是一个由备份集编号和建立时间压缩后组成的 8 字符名称。利用%u 可以为每个备份集产生一个唯一的名称;

%p：表示备份集中的备份片的编号，从 1 开始编号;

%U：是%u_%p_%c 的简写形式，利用它可以为每一个备份片段(既磁盘文件)生成一个唯一的名称，这是最常用的命名方式;

%t：备份集时间戳;

%T:年月日格式(YYYYMMDD);

channel 的概念：一个 channel 是 rman 于目标数据库之间的一个连接，"allocate channel"命令在目标数据库启动一个服务器进程，同时必须定义服务器进程执行备份和恢复操作使

用的 I/O 类型

通道控制命令可以用来:

控制 rman 使用的 OS 资源

影响并行度

指定 I/O 带宽的限制值(设置 limit read rate 参数)

指定备份片大小的限制(设置 limit kbytes)

指定当前打开文件的限制值(设置 limit maxopenfiles)

==========RMAN 一周典型备份方案==========

1.星期天晚上 -level 0 backup performed(全备份)
2.星期一晚上 -level 2 backup performed
3.星期二晚上 -level 2 backup performed
4.星期三晚上 -level 1 backup performed
5.星期四晚上 -level 2 backup performed
6.星期五晚上 -level 2 backup performed
7.星期六晚上 -level 2 backup performed

如果星期二需要恢复的话，只需要 1+2,
如果星期四需要恢复的话，只需要 1+4,
如果星期五需要恢复的话，只需要 1+4+5,
如果星期六需要恢复的话，只需要 1+4+5+6.

自动备份:备份脚本+crontab
bakl0
bakl1
bakl2

执行脚本:
rman target / msglog=bakl0.log cmdfile=bakl0 (/表示需要连接的目标数据库,msglog 表示日志文件，cmdfile 表示的是脚本文件)
rman target / msglog=bakl1.log cmdfile=bakl1
rman target / msglog=bakl2.log cmdfile=bakl2

实例：rman target system/oracle@ora10g(/) msglog=/u01/rmanbak/bakl1.log cmdfile=/u01/rmanbak/bakl0

完整的命令:/u01/oracle/product/10.2.0/bin/rman target system/oracle@ora10g(/) msglog=/u01/rmanbak/bakl1.log cmdfile=/u01/rmanbak/bakl0

把备份脚本放到/u01/rmanbak/script 目录下面,vi bakl0,bakl0 的内容为:

run{

```
allocate channel cha1 type disk;
backup
incremental level  0
format '/u01/rmanbak/inc0_%u_%T'(u 表示唯一的 ID,大 T 是日期，小 t 是时间)
tag monday_inc0 //标签可以顺便起，没关系
database;
release channel cha1;
}
```

，类似就可以写出 bakl1,bakl2 相应的脚本.

自动备份

crontab

crontab -e -u oracle(改命令的意思是编辑 oracle 用户的定时执行(-e,edit -u oracle,oracle 用户))

分 时 日 月 星期(0 代表星期天)

45 23 * * 0 rman target / msglog=bakl0.log cmdfile=bakl0(星期天的 23:45 会以 oracle 用户的身份来执行命令)

45 23 * * 1 rman target / msglog=bakl2.log cmdfile=bakl2

45 23 * * 2 rman target / msglog=bakl2.log cmdfile=bakl2

45 23 * * 3 rman target / msglog=bakl1.log cmdfile=bakl1

45 23 * * 4 rman target / msglog=bakl2.log cmdfile=bakl2

45 23 * * 5 rman target / msglog=bakl2.log cmdfile=bakl2

45 23 * * 6 rman target / msglog=bakl2.log cmdfile=bakl2

然后启动 crontab ,启动 crontab 的命令:

root> service crond restart

✓ RMAN 恢复实例

在非 catalog 模式下，备份的信息存储在 controlfile 文件中，如果 controlfile 文件发生毁坏，那么就不能能够进行恢复，

使用在备份的时候需要把 controlfile 也进行自动备份

```
RMAN>show all;
using target database control file instead of recovery catalog
RMAN configuration parameters are:
CONFIGURE RETENTION POLICY TO REDUNDANCY 1; # default
CONFIGURE BACKUP OPTIMIZATION OFF; # default
CONFIGURE DEFAULT DEVICE TYPE TO DISK; # default
CONFIGURE CONTROLFILE AUTOBACKUP OFF; # default
CONFIGURE CONTROLFILE AUTOBACKUP FORMAT FOR DEVICE TYPE DISK TO '%F'; # default
CONFIGURE DEVICE TYPE DISK PARALLELISM 1 BACKUP TYPE TO BACKUPSET; # default
```

CONFIGURE DATAFILE BACKUP COPIES FOR DEVICE TYPE DISK TO 1; # default

CONFIGURE ARCHIVELOG BACKUP COPIES FOR DEVICE TYPE DISK TO 1; # default

CONFIGURE MAXSETSIZE TO UNLIMITED; # default

CONFIGURE ENCRYPTION FOR DATABASE OFF; # default

CONFIGURE ENCRYPTION ALGORITHM 'AES128'; # default

CONFIGURE ARCHIVELOG DELETION POLICY TO NONE; # default

CONFIGURE SNAPSHOT CONTROLFILE NAME TO '/home/oracle/product/10.20/dbs/snapcf_ora10g.f'; # default

其中 CONFIGURE CONTROLFILE AUTOBACKUP OFF; 没有对 controlfile 进行 autobackup,使用我们需要运行下面命令来对 controlfile 进行自动备份

RMAN> CONFIGURE CONTROLFILE AUTOBACKUP ON;

RMAN> show all;

手动备份控制文件：

backup current controlfile

Dbid 表示 database 的一个 ID，将来用于恢复 spfile 和 controlfile 时候要用到.

RMAN> connect target /

connected to target database: ORA10G (DBID=3988862108)

这个 Dbid=3988862108

RMAN> list backup;查看以前备份的信息

RMAN>delete backupset 24;//24 代表 backupset 的编号

RMAN>backup format '/u01/rmanbak/full_%T_%U.bak' database plus archivelog;(进行一次全备份)

验证备份：

RMAN> validate backupset 3; //3 代表 backupset 的编号

口令文件丢失(不属于 rman 备份的范畴),我们只需要用一个命令来重建这个文件就可以了:

orapw file=orapwsid password=pass entries=5; //口令文件的路径:/u01/oracle/product/10.20/db_1/dbs 目录下

oracle> cd /u01/oracle/product/10.20/db_1/dbs

oracle> rm orapwora10g;(文件删除，模拟丢失)

oracle> orapwd file=orapwora10g password=oracle entries=5;(重新建立一个文件),entries 的意思(DBA 的用户最多有 5 个)

SPFILE 丢失:

startup nomount;

set dbid 3988862108;

restore spfile from autobackup;

shutdown immediate;

set dbid 3988862108;

startup;

模拟操作:

oracle> mv spfileora10g.ora spora10g.ora

oracle>rman target /;

rman> shutdown immediate;

rman> startup nomount;

startup failed: ORA-01078: failure in processing system parameters

LRM-00109: could not open parameter file '/home/oracle/product/10.20/dbs/initora10g.ora'

rman>set dbid 3988862108;

rman>restore spfile from autobackup;

执行该命令，如果没有找到的话，那可能是文件的路径发生错误.可以通过直接赋予它的文件

rman>restore spfile from '/u01/oracle/flash_recovery_area/ORA10G/autobackup/2008_12_09/o1_mf_s_673025706_4mw7xc79_.bkp

在 dbs/目录下产生 spfileora10g.ora 文件。证明 spfile 已经恢复好

rman> shutdown immediate;

rman> startup ;(如果该命令不能够启动数据库，那么需要 set dbid 3988862108)

controlfile 丢失:

startup nomount;

restore controlfile from autobackup;

alter database mount;

recover database;

alter database open resetlogs;

注意:在做了 alter database open resetlogs;会把 online redelog file 清空，数据文件丢失.所以这个时候要做一个全备份。

oracle>rm *.ctl

oracle>rman target / ;//不能够连接到 rman ,因为 controlfile 丢失

oracle>sqlplus /nolog;

SQL>shutdown immediate; //因为 controlfile 丢失，不能够正常 shutdown

SQL>shutdown abort;

oracle>rman target /;

rman>startup nomount;

rman>restore controlfile from autobackup;

rman>alter database mount;

rman>alter database open resetlogs;

RMAN-00571: ===

RMAN-00569: =============== ERROR MESSAGE STACK FOLLOWS ===============

RMAN-00571: ===

RMAN-03002: failure of alter db command at 12/09/2008 16:21:13

ORA-01194: file 1 needs more recovery to be consistent

ORA-01110: data file 1: '/home/oracle/oradata/ora10g/system01.dbf

//出错, redo log 的 scn 记录在 controlfile 里面的，因为我们有新的 controlfile,所以需要 resetlogs;

/*

resetlogs 命令表示一个数据库逻辑生存期的结束和另一个数据库逻辑生存期的开始，每次使用 resetlogs 命令的时候，SCN 不会被重置，不过 oracle 会重置日志序列号，而且会重置

联机重做日志内容.

这样做是为了防止不完全恢复后日志序列会发生冲突（因为现有日志和数据文件间有了时间差）。

*/

rman>recover database;

rman>alter database open resetlogs;

Redolog file 丢失:(下面的这些语句一定要在 sqlplus 中执行,不是在 rman 中执行)

(sqlplus/nolog)

1.shutdown immediate;

2.startup mount;

3.recover database until cancel;(media recovery)

4.alter database resetlogs;

数据文件丢失(在 rman 中执行 sql 语句，在 sql 后面用双引号括起来):

1. sql "alter database datafile 3 offline";

2. restore datafile 3

3. recover datafile 3

4. sql "alter database datafile 3 online";

表空间丢失:

1. sql "alter tablespace users offline";//如果文件不存在，则用 sql "alter tablespace users offline immeidate";

2. restore tablespace users;

3. recover tablespace users; //与 online redolog file 信息一致

4. sql "alter tablespace users online";

非 catalog 方式完全恢复

数据库出现问题:

1.startup nomount;
2.restore controlfile from autobackup;
3.alter database mount;
4.restore database;
5.recover database;
6.alter database open resetlogs;

模拟操作:
oracle ora10g> rm *;
oracle ora10g> ls;
oracle ora10g> //数据文件，控制文件全部删除

oracle ora10g> rman target /; //因为 controlfile 丢失，不能够连接到 rman
oracle ora10g> sqlplus /nolog;
oracle ora10g> connect / as sysdba;
oracle ora10g> shutdown abort;
oracle ora10g> rman target /

rman> startup nomount;
rman> restore controlfile from autabackup;
rman> alter database mount;
rman> restore database;
rman> recover database; //online redolog 不存在

SQL>recover database until cancel; //当 redo log 丢失，数据库在缺省的方式下，是不容许进行 recover 操作的,那么如何在这种情况下操作呢
SQL>create pfile from spfile;

vi /u01/product/10.20/dbs/initora10g.ora，在这个文件的最后一行添加
*.allow_resetlogs_corruption='TRUE'; //容许 resetlog corruption

SQL>shutdown immediate;
SQL>startup pfile='/u01/product/10.20/dbs/initora10g.ora' mount;
SQL>alter database open resetlogs;

基于时间点的恢复:
```
run{
  set until time "to_date(07/01/02 15:00:00','mm/dd/yy hh24:mi:ss')";
  restore database;
  recover database;
  alter database open resetlogs;
```

}

ALTER SESSION SET NLS_DATE_FORMAT='YYYY-MM-DD HH24:MI:SS';

1.startup mount;

2.restore database until time "to_date('2009-7-19 13:19:00','YYYY-MM-DD HH24:MI:SS')";

3.recover database until time "to_date('2009-7-19 13:19:00','YYYY-MM-DD HH24:MI:SS')";

4.alter database open resetlogs;

如果有 open resetlogs,都是不完整恢复.

基于 SCN 的恢复:

1.startup mount;

2.restore database until scn 10000;

3.recover database until scn 10000;

4.alter database open resetlogs;

基于日志序列的恢复:

1.startup mount;

2.restore database until SEQUENCE 100 thread 1; //100 是日志序列

3.recover database until SEQUENCE 100 thread 1;

4.alter database open resetlogs;

日志序列查看命令： SQL>select * from v$log;其中有一个 sequence 字段.resetlogs 就会把 sequence 置为 1

==========RMAN catalog 模式下的备份与恢复==========

1.创建 Catalog 所需要的表空间

SQL>create tablespace rman_ts size datafile '/u01/oracle/oradata/ora10g/rmants.dbf' 20M;

2.创建 RMAN 用户并授权

SQL>create user rman identified by rman default tablespace rman_ts quota unlimited on rman_ts;

SQL>grant recovery_catalog_owner to rman;(grant connect to rman)

查看角色所拥有的权限: select * from dba_sys_privs where grantee='RECOVERY_CATALOG_OWNER';

(RECOVER_CATALOG_OWNER,CONNECT,RESOURCE)

3.创建恢复目录

oracle>rman catalog rman/rman

RMAN>create catalog tablespace rman_ts;

RMAN>register database;(database 是 target database)

database registered in recovery catalog

starting full resync of recovery catalog

full resync complete

RMAN> connect target /;

以后要使用备份和恢复，需要连接到两个数据库中,命令:

oracle>rman target / catalog rman/rman (第一斜杠表示 target 数据库，catalog 表示 catalog 目录 rman/rman 表示 catalog 用户名和密码)

命令执行后显示：

Recovery Manager: Release 10.2.0.1.0 - Production on Wed Dec 10 15:00:42 2008

Copyright (c) 1982, 2005, Oracle. All rights reserved.

connected to target database: ORA10G (DBID=3988862108)

connected to recovery catalog database

命令解释:

Report schema Report shema 是指在数据库中需找 schema

List backup 从 control 读取信息

Crosscheck backup 看一下 backup 的文件，检查 controlfile 中的目录或文件是否真正在磁盘上

Delete backupset 24 24 代表 backupset 的编号，既 delete 目录，也 delete 你的文件

注意:在做了 alter database open resetlogs;会把 online redelog file 清空，数据文件丢失.所以这个时候要做一个全备份。

resetlogs 命令表示一个数据库逻辑生存期的结束和另一个数据库逻辑生存期的开始，每次使用 resetlogs 命令的时候，SCN 不会被重置，不过 oracle 会重置日志序列号，而且会重置

联机重做日志内容.这样做是为了防止不完全恢复后日志序列会发生冲突（因为现有日志和数据文件间有了时间差）。

Rman 归档文件丢失导致不能备份的，在备份前先执行以下两条命令

crosscheck archivelog all;

delete expired archivelog all;

✓ Oracle Rman 命令详解(List report backup configure)

一、list 常用命令总结备忘

list incarnation;

list backup summary;

list backup of database summary;

list backup of tablespace summary;

list backup of datafile n,n summary;

list archivelog all summary;

list backup by file;

list backup;

list expired backup;

list copy;

list backup of spfile;

list backup of controlfile;

list backup datafile n,n,n;

list backup tablespace tablespace_name;

list backup of archivelog all;

list backup of archivelog from scn ...;

list backup of archivelog until scn ...;

list backup of archivelog from sequence ..;

list backup of archivelog until time 'sysdate-10';

list backup of archivelog {all, from, high, like, logseq, low, scn, sequence, time, until};

1. List 当前 RMAN 所备份的数据库：

RMAN> list incarnation;

汇总查询：--如果备份文件多的话多用这两个 list 命令可以对备份文件有个总体了解。

1.1. list backup summary; --概述可用的备份

B 表示 backup

A 表示 Archivelog、 F 表示 full backup、 0,1,2 表示 incremental level 备份

A 表示可用 AVALIABLE、 X 表示 EXPIRED

这个命令可以派生出很多类似命令，例如

list backup of database summary

list backup of archivelog all summary

list backup of tablespace users summary;

list backup of datafile n,n,n summary

这些命令可以让我们对已有的备份文件有一个整体，直观的了解。

1.2.list backup by file;--按照文件类型分别列出

分别为：数据文件列表、归档日志列表、控制文件列表、SPFILE 列表

1.3.list backup;

这个命令列出已有备份集的详细信息。

1.4.list expired backup;

列出过期的备份文件

1.5.list copy;

列出 copy 文件

list copy of database;

list copy of controlfile;

list copy of tablespace users;

list copy of datafile n,n,n;

list copy of archivelog all;

list copy of archivelog from scn 10000;

list copy of archivelog until sequence 12;

2. List 相关文件的信息

list backup of {archivelog, controlfile, database, datafile, spfile, tablespace};

list backup of database; --full,incremental,tablespace,datafile

2.1 服务器参数文件:

list backup of spfile;

2.2 控制文件：

list backup of controlfile;

2.3 数据文件：

list backup of datafle n,n,n,n;

2.4 表空间：

list backup of tablespace tablespace_name;--表空间对应的 backup

2.5 归档日志：

list backup of archivelog {all, from, high, like, logseq, low, scn, sequence, time, until};

list backup of archivelog all;

list backup of archivelog until time 'sysdate-1';

list backup of archivelog from sequence 10;

list backup of archivelog until sequence 10;

list backup of archivelog from scn 10000;

list backup of archivelog until scn 200000;

list archivelog from scn 1000;

list archivelog until scn 2000;

list archivelog from sequence 10;

list archivelog until sequence 12;

二、report 常用命令总结备忘

report 用于判断数据库当前可恢复状态、以及数据库已有备份的信息。

最常使用的是 report obsolete; report schema;

report {device, need, obsolete, schema, unrecoverable}

report schema;

report obsolete;

report unrecoverable;

report need backup;

report need backup days=3; --报告最近 3 天内没有备份的文件

report need backup redundancy=3; --报告冗余次数小于 3 的数据文件。

report need backup recovery window of 2 days;

2.1.report schema;

报告数据库模式

22.report obsolete;

报告已丢弃的备份集(配置了保留策略)。

2.3.report unrecoverable;

报告当前数据库中不可恢复的数据文件(即没有这个数据文件的备份、或者该数据文件的备份已经过期)

2.4.report need backup;

报告需要备份的数据文件(根据条件不同)

report need backup days=3;

--最近三天没有备份的数据文件(如果出问题的话，这些数据文件将需要最近 3 天的归档日志才能恢复)

report need backup incremental=3;

--需要多少个增量备份文件才能恢复的数据文件。(如果出问题，这些数据文件将需要 3 个增量备份才能恢复)

report need backup redundancy=3;

--报告出冗余次数小于 3 的数据文件

--例如数据文件中包含 2 个数据文件 system01.dbf 和 users01.dbf.

--在 3 次或都 3 次以上备份中都包含 system01.dbf 这个数据文件，而 users01.dbf 则小于 3 次

--那么，报告出来的数据文件就是 users01.dbf

--即，报告出数据库中冗余次数小于 n 的数据文件

report need backup recovery window of 2 days;

--报告出恢复需要 2 天归档日志的数据文件

三、backup 常用命令总结备忘

1.设置备份标记

backup database tag='full_bak1';

注：每个标记必须唯一，相同的标记可以用于多个备份只还原最新的备份。

2.设置备份集大小(一次备份的所有结果为一个备份集，要注意备份集大小)

backup database maxsetsize=100m tag='datafile1';

注：maxsetsize 限定备份集的大小。所以必须大于数据库总数据文件的大小，否则会报错。

RMAN-06183: datafile or datafile copy larger than MAXSETSIZE: file# 1 /data/oradata/system01.dbf

3.设置备份片大小(磁带或文件系统限制)

```
run {
allocate channel c1 type disk maxpicecsize 100m format '/data/backup/full_0_%U_%T';
backup database tag='full_0';
release channel c1;
}
```

可以在 allocate 子句中设定每个备份片的大小，以达到磁带或系统限制。

也可以在 configure 中设置备份片大小。

```
Configure channel device type disk maxpiecesize 100 m;
configure channel device type disk clear;
```

4.备份集的保存策略

```
backup database keep forever;                          --永久保留备份文件
backup database keep until time='sysdate+30'; --保存备份 30 天
```

5.重写 configure exclude 命令

```
backup databas noexclude keep forever tag='test backup';
```

6.检查数据库错误

```
backup validate database;
```

使用 RMAN 来扫描数据库的物理/逻辑错误，并不执行实际备份。

7.跳过脱机，不可存取或只读文件

```
backup database skip readonly;
backup database skip offline;
backup database skip inaccessible;
backup database ship readonly skip offline ship inaccessible;
```

8.强制备份

```
backup database force;
```

9.基于上次备份时间备份数据文件

1>只备份添加的新数据文件

```
backup database not backed up;
```

2>备份"在限定时间周期内"没有被备份的数据文件

```
backup database not backed up since time='sysdate-2';
```

10.备份操作期间检查逻辑错误

```
backup check logical database;
backup validate check logical database;
```

11.生成备份副本

backup database copies=2;

12.备份控制文件

backup database device type disk includ current controlfile;

四、configure 常用命令总结备忘

4.1 显示当前的配置信息

1.1 RMAN> show all;

RMAN 配置参数为:

CONFIGURE RETENTION POLICY TO REDUNDANCY 1; # default

CONFIGURE BACKUP OPTIMIZATION OFF; # default

CONFIGURE DEFAULT DEVICE TYPE TO DISK; # default

CONFIGURE CONTROLFILE AUTOBACKUP OFF; # default

CONFIGURE CONTROLFILE AUTOBACKUP FORMAT FOR DEVICE TYPE DISK TO '%F'; # default

CONFIGURE DEVICE TYPE DISK PARALLELISM 1 BACKUP TYPE TO BACKUPSET; # default

CONFIGURE DATAFILE BACKUP COPIES FOR DEVICE TYPE DISK TO 1; # default

CONFIGURE ARCHIVELOG BACKUP COPIES FOR DEVICE TYPE DISK TO 1; # default

CONFIGURE MAXSETSIZE TO UNLIMITED; # default

CONFIGURE ENCRYPTION FOR DATABASE OFF; # default

CONFIGURE ENCRYPTION ALGORITHM 'AES128'; # default

CONFIGURE ARCHIVELOG DELETION POLICY TO NONE; # default

CONFIGURE SNAPSHOT CONTROLFILE NAME TO 'D:\ORACLE\PRODUCT\10.2.0\DB_1\DATABASE\S

NCFDBA.ORA'; # default

1.2 查询 RMAN 设置中非默认值:

SQL> select name,value from v$rman_configuration;

4.2. 常用的 configure 选项

4.2.1 保存策略 (retention policy)

configure retention policy to recovery window of 7 days;

configure retention policy to redundancy 5;

configure retention policy clear;

CONFIGURE RETENTION POLICY TO NONE;

第一种 recover window 是保持所有足够的备份，可以将数据库系统恢复到最近七天内的任意时刻。任何超过最近七天的数据库备份将被标记为 obsolete。

第二种 redundancy 是为了保持可以恢复的最新的 5 份数据库备份，任何超过最新 5 份的备份都将被标记为 redundancy。它的默认值是 1 份。

第三四：NONE 可以把使备份保持策略失效，Clear 将恢复默认的保持策略

一般最安全的方法是采用第二种保持策略。

4.2.2 备份优化 backup optimization

configure backup optimization on;

configure backup optimization off;

configure backup optimization clear;

默认值为关闭，如果打开，rman 将对备份的数据文件及归档等文件进行一种优化的算法。

4.2.3 默认设备 default device type

configure default device type to disk;

configure default device type to stb;

configure default device type clear;

是指定所有 I/O 操作的设备类型是硬盘或者磁带，默认值是硬盘

磁带的设置是 CONFIGURE DEFAULT DEVICE TYPE TO SBT;

4.3.4 控制文件 controlfile

configure controlfile autobackup on;

configure controlfile autobackup format for device type disk to '/cfs01/backup/conf/conf_%F';

configure controlfile autobackup clear;

configrue controlfile autobackup format for device type disk clear;

configrue snapshot controlfile name to '/cfs01/backup/snapcf/scontrofile.snp';

--是配置控制文件的快照文件的存放路径和文件名，这个快照文件是在备份期间产生的，用于控制文件的读一致性。

configrue snapshot controlfile name clear;

强制数据库在备份文件或者执行改变数据库结构的命令之后将控制文件自动备份，默认值为关闭。这样可以避免控制文件和 catalog 丢失后，控制文件仍然可以恢复。

4.3.5 并行数(通道数) device type disk|stb pallelism n;

configure device type disk|stb parallelism 2;

configure device type disk|stb clear; --用于清除上面的信道配置

configure channel device type disk format 'e/:rmanback_%U';

configure channel device type disk maxpiecesize 100m

configure channel device type disk rate 1200K

configure channel 1 device type disk format 'e/:rmanback_%U';

configure channel 2 device type disk format 'e/:rmanback_%U';

configure channel 1 device type disk maxpiecesize 100m

配置数据库设备类型的并行度。

4.3.6 生成备份副本 datafile|archivelog backup copies

configure datafile backup copies for device type disk|stb to 3;

configure archivelog backup copies for device type disk|stb to 3;

--是设置数据库的归档日志的存放设备类型

configure datafile|archivelog backup copies for device type disk|stb clear

BACKUP DEVICE TYPE DISK DATABASE

FORMAT '/disk1/backup/%U', '/disk2/backup/%U', '/disk3/backup/%U';

是配置数据库的每次备份的 copy 数量，oracle 的每一次备份都可以有多份完全相同的拷贝。

4.3.7 排除选项 exclude

configure exclude for tablespace 'users';

configrue exclude clear;

此命令用于将指定的表空间不备份到备份集中， 此命令对只读表空间是非常有用的。

4.3.8 备份集大小 maxsetsize

configure maxsetsize to 1G|1000M|1000000K|unlimited;

configure maxsetsize clear;

4.3.9 其它选项 auxiliary

CONFIGURE AUXNAME FOR DATAFILE 1 TO '/oracle/auxfiles/aux_1.f';

CONFIGURE AUXNAME FOR DATAFILE 2 TO '/oracle/auxfiles/aux_2.f';

CONFIGURE AUXNAME FOR DATAFILE 3 TO '/oracle/auxfiles/aux_3.f';

CONFIGURE AUXNAME FOR DATAFILE 4 TO '/oracle/auxfiles/aux_4.f';

-

CONFIGURE AUXNAME FOR DATAFILE 1 CLEAR;

CONFIGURE AUXNAME FOR DATAFILE 2 CLEAR;

CONFIGURE AUXNAME FOR DATAFILE 3 CLEAR;

CONFIGURE AUXNAME FOR DATAFILE 4 CLEAR;

Rman 的 format 格式中的%

%c 备份片的拷贝数

%d 数据库名称

%D 位于该月中的第几天 (DD)

%M 位于该年中的第几月 (MM)

%F 一个基于 DBID 唯一的名称,这个格式的形式为 c-IIIIIIIII-YYYYMMDD-QQ,其中 IIIIIIIII 为该数据库的 DBID，YYYYMMDD 为

日期，QQ 是一个 1-256 的序列

%n 数据库名称，向右填补到最大八个字符

%u 一个八个字符的名称代表备份集与创建时间

%p 该备份集中的备份片号，从 1 开始到创建的文件数

%U 一个唯一的文件名，代表%u_%p_%c

%s 备份集的号

%t 备份集时间戳

%T 年月日格式(YYYYMMDD)

- ✓ **Oracle Rman 跨 resetlogs 版本恢复**

http://blog.csdn.net/tianlesoftware/archive/2009/10/17/4682463.aspx

- ✓ **ORACLE 数据库逻辑备份 简单 EXP/IMP**

http://blog.csdn.net/tianlesoftware/archive/2009/10/24/4718366.aspx

- ✓ **Oracle 10g EXPDP 和 IMPDP 使用说明**

http://blog.csdn.net/tianlesoftware/archive/2009/10/16/4674224.aspx

3. **三．恢复**

✓ 理论知识：

Oracle 在运行过程中，所有对于数据的修改都是在内存中进行，Oracle 每要修改一个记录必须先把记录所在的数据块加载到内存中，然后在内存中进行修改。但是提交（commit）时，修改的数据块不会立即写回磁盘。基于性能考虑，Oracle 是采用“延时写”的算法定期批量的把数据块写回磁盘。因此在数据库运行过程中，内存的内容总是比磁盘数据新。 当数据库正常关闭时（Shutdown Immediate,shutdown normal,shutdown tracsactional）,Oracle 会把 SGA 内容全部写回磁盘后才关闭数据库，这时内存和磁盘就完全同步了。 所以正常关闭数据库后数据不会丢失，但是如果数据库是异常关闭（突然短线，shutdown abort），内存中的数据来不及同步到磁盘，这是就会产生了数据不一致，Oracle 在次打开数据库时，就需要进行实例恢复。

Oracle 的 Redo 机制保证了数据库恢复的可行性，在修改数据之前，代表本次修改操作的 Redo 记录必须先被保存下来（Write Ahead Logging），然后才真正修改数据记录。在处理 commit 语句时，Oracle 会在 Log buffer 产生一条 commit 记录，为了保证事务的持久化，所有 Redo 记录和这一条 commit 记录都要被写到磁盘的联机日志文件（Log Force At commit），但是数据块（Data Block）不必写回磁盘。如果联机日志空间不够，还会触发日志切换（Log Switch），旧日志的检查点必须完成才能被覆盖，如果采用归档模式，这个日志还必须完成归档才能覆盖。

这些日志中都会带有 SCN，SCN 类似于时间戳，Oracle 按照 SCN 对日志内容进行排序，就可以得到操作历史，Oracle 也是根据 SCN 来判断数据文件是否需要恢复的。

数据库在正常运行时，每个数据文件的终止 SCN(STOP SCN)会被设置为无穷大（NULL），而其他的那些 SCN 应该完全一样。如果数据库正常关闭，关闭之前会执行一个检查点动作，每个数据文件的终止 SCN 会被设置成启动 SCN（Start SCN）。如果数据库异常关闭，终止 SCN 来不及设置为启动 SCN，仍然保持 NULL。

当 clean shutdown 时，checkpoint 会进行，并且此时 datafile 的 stop scn 和 start scn 会相同。 等到我们开启数据库时，Oracle 检查 datafile header 中的 start scn 和存于 control file 中的 datafile 的 scn 是否相同， 如果相同，接着检查 start scn 和 stop scn 是否相同，如果仍然相同，数据库就会正常开启，否则就需要 recovery... 等到数据库开启后，储存在 control file 中的 stop scn 就会恢复为 NULL 值，此时表示 datafile 是 open 在正常模式下了。

如果不正常 SHUTDOWN (shutdown abort)，则 mount 数据库后，你会发现 stop scn 并不是等于其它位置的 scn， 而是等于 NULL，这表示 Oracle 在 shutdown 时没有进行 checkpoint，下次开机必须进行 crash recovery。

注意：当发生 checkpoint 时，会把 SCN 写到四个地方去。三个地方于 control file 内，一个在 datafile header。

Control file 三个地方为

1.System checkpoint SCN

2.2.Datafile checkpoint SCN

3.Stop SCN

另外一个地方在 datafile header 内

4.Start SCN

Oracle 恢复可以分成实例恢复（Instance Recovery）,介质恢复（Media Recovey），其中介质恢复又可分为完全恢复（Complete Recovery）和不完全恢复（Incomplete Recovery）。

✓ 恢复种类

3.2.1 Instance Recovery -- 由 Oracle 自动完成，无需 DBA 干预

如果实例异常关闭（宕机，shutdown abort），并且数据文件，控制文件，联机日志都没有丢失。在下次启动时，要利用联机日志的内容进行恢复，这种恢复就是实例恢复（Instance Recovery）。

Instance Recovery 主要包括 3 个阶段：

1） 根据联机日志内容进行 Rollover。

2） 打开数据库，提供服务

3） SMON 或者用户进程进行 Rollback。

3.2.2 Media Recovery

如果发生数据文件丢失或者破坏，就需要使用备份和归档日志来进行恢复， 这种恢复就是 介质恢复，它需要有备份，归档日志，联机日志一起才能完成。又分为 安全恢复和不完全恢复两种。

3.2.3 Crash Recovery -- 由 Oracle 自动完成。

Instance Recovery 是实例发生 Crash 后进行的 Recovery，这种恢复是在故障节点进行，而 RAC 中的 Crash Recovery 是某个实例发生 Crash 后在其他实例上进行的 Recovery。 这种 Recovery 有一个特殊要求：在健康节点执行 Crash

Recovery 时，必须要保证故障节点不能在对共享数据进行操作，也就是要对故障节点进行 IO 隔离（IO Fencing），这是由 CSS 服务来保证的。

在 Crash Recovery 过程中 PCM Lock 起到了重要作用，恢复实例（执行 Recovery 动作的实例）根据数据块的 PCM-Lock 状态来决定数据块是否需要进行恢复。

Crash Recovery 可分为 3 个阶段：

1）First-Pass Log Read

2）Recovery Claim Locking

3）Second-Pass Log Read

3.2.4 Online Block Recovery

Online Block Recovery 是 RAC 所特有的， 如果某个用户进程在修改数据时异常死掉，导致 SGA 的 Data buffer 数据不一致，或者说 Data Buffer 被破坏，这时就会触发 Online Block Recovery，这个动作可有 PMON 进程或者前台进程完成。这个恢复过程需要一个恢复起点， 这个起点就是最近的 Past Image。

✓ 介质恢复

介质恢复指磁盘介质发生损坏，导致数据文件无法访问，这时必须利用备份文件在新的磁盘上恢复出数据文件。

3.3.1 完全恢复

它是把数据库恢复到发生故障时的状态，名字中的完全指没有任何数据损失，要实现这个目标，必须满足一定的条件： 备份，从备份之后的所有归档日志，联机日志都可用。

完全恢复是最简单的一种恢复， 只需要两个命令： restore database 和 recover database。

✓ RMAN 备份与恢复 实例

参见第二章：备份

✓ **补充**

在次补充一点知识： RECOVER DATABASE UNTIL CANCEL 和 RECOVER DATABASE UNTIL CANCEL USING BACKUP CONTROLFILE 区别

1） RECOVER DATABASE UNTIL CANCEL
==> DATAFILE HEADER SCN 一定会小于 CONTROLFILE 的 DATAFILE SCN
如果你有进行 RESTORE DATAFILE，则该 RESTORE 的 DATAFILE HEADER SCN 一定会小于目前 CONTROLFILE 的 DATAFILE SCN，此时会无法开启数据库，必须进行 media recovery。重做 archive log 直到该 datafile header 的 SCN=current scn

2） RECOVER DATABASE UNTIL CANCEL USING BACKUP CONTROLFILE;

==> DATAFILE HEADER SCN 一定会大于 CONTROLFILE 的 DATAFILE SCN

如果只是某 TABLE 被 DROP 掉，没有破坏数据库整体数据结构，还可以用 NCOMPLETE RECOVERY 解决 如果是某个 TABLESPACE OR DATAFILE 被 DROP 掉，因为档案结构已经破坏，目前的 CONTROL FILE 内已经没有 该 DATAFILE 的信息，就算你只 RESTORE DATAFILE 然后进行 INCOMPLETE RECOVERY 也无法救回被 DROP 的 DATA FILE。

只好 RESOTRE 之前备份的 CONTROL FILE(里头被 DROP DATAFILE Metadata 此时还存在)，不过 RESTOREC CONTROL FILE 后 此时 Oracle 会发现 CONTROL FILE 内的 SYSTEM SCN 会小于目前的 DATAFILE HEADER SCN，也不等于目前储存于 LOG FILE 内的 SCN， 此时就必须使用 RECOVER DATABASE UNTIL CANCEL USING BACKUP CONTROLFILE 到 DROP DATAFILE OR DROP TABLESPACE 之前的 SCN。

3.3.2 不完全恢复

不完全恢复是指数据库无法恢复到发生故障那一点的状态，而只能恢复到之前一段时间的状态，这就以为着承受一定量的数据损失。

Oracle 运行时包括参数文件，控制文件，数据文件，联机日志，那么哪些文件会导致不完全恢复呢？

参数文件只是一个文本文件，丢失没有关系。控制文件通常有多个文件相互冗余。而且在做全库备份时，控制文件会被自动备份，故所有文件都损坏，也可以通过备份进行恢复，即使没有备份，也可以通过重建控制文件来恢复，也不会造成数据丢失。如果是数据文件损坏，只要有备份和备份后的完整的日志文件，也可以完成恢复，不会造成数据丢失。

联机日志比较特殊，通过前面的介绍，我们知道在数据库异常关机的情况下，它可能造成数据丢失。

我们来看一下联机日志损坏的恢复方法：

先用 SQL 查看一下出问题的联机日志是什么状态：

SQL> Select thread#,group#,status from v$log;

1）如果是 Inactive 状态的联机日志，因为它里面的记录已经同步到数据文件，所以只需要把该日志删掉即可。

2）如果是 Active/ current 状态的连接日志， 因为他们里面有记录没有同步到数据文件，可以通过如下方式来恢复：

（1）关闭所有实例

（2）在受损实例上，启动到 mount 状态

（3）执行 alter database open resetlogs

（4）如果在第三步出现错误，并其实需要不完全恢复，就执行一下： recover database until cancel

（5）实例启动成功后，启动其他实例

（6）立即对数据库进行一次全备。

说明： 在做了 alter database open resetlogs;会把 online redelog file 清空，数据文件丢失.所以这个时候要做一个全备份。resetlogs 命令表示一个数据库逻辑生存期的结束和另一个数据库逻辑生存期的开始，每次使用 resetlogs 命令的时候，SCN 不会被重置，不过 oracle 会重置日志序列号，而且会重置联机重做日志内容.这样做是为了防止不完全恢复后日志序列会发生冲突（因为现有日志和数据文件间有了时间差）。

✓ 时间点恢复

startup mount
alter session set nls_date_format='yyyy-mm-dd hh24:mi:ss';
recover database until time '2005-01-17 11:57:28';
恢复到提交完成时刻
11:58:33 SQL> alter database open resetlogs;

20. 简单的 EXP/IMP

1. 具备知识

1. 逻辑备用数据库的优点

逻辑备用数据库提供了健壮、有效的灾难恢复和高可用性解决方案。当主数据库因不可预见的故障出现失败时，逻辑备用数据库可以立即转换为主数据库，从而减少了主数据库的停机时间。

逻辑备用数据库提供了高级别的数据保护措施。通过使用 Data Guard 的最大保护模式，主数据库事务变化会同步传送到逻辑备用数据库。这样，当主数据库因不可预见的故障出现失败时，可以确保不会丢失数据。

除了可以满足灾难恢复需求之外，逻辑备用数据库还可以用于其他商业目的。逻辑备用数据库可以拥有自身的数据库方案，并且数据库用户可以在这些方案上执行 DDL 或 DML 操作。

降低主数据为听工作负载。与主数据库一样，逻辑备用数据为可以一直处于 OPEN 状态。如果主数据库负载量很大，那么可以将数据统计、数据报表和执行查询等操作转移到逻辑备用数据库来完成，从而节省主数据库的 CPU 和 I/O 开销。

2. 逻辑备用数据库支持的数据类型和表存储属性。

字符串类型：CHAR、VARCHAR2、VARCHAR、NCHAR、NVARCHAR2 和 LONG。

数字类型：NUMBER、BINARY_FLOAT、BINARY_DOUBLE。

日期时间类型：DATE、TIMESTAMP、TIMESTAMP WITH TIME ZONE、TIMESTAMP WITH LOCAL ZONE、INTERVAL YEAR TO MONTH、INTERVAL DAY TO SECOND。

二进制类型：RAW、LONG RAW。

大对象类型：CLOB、BLOB、NCLOB。

3. 逻辑备用数据库不支持的数据类型。

BFILE

ROWID、UROWID

用户自定义类型(CREATE TYPE)、REF 对象类型、VARRAY、嵌套表、XMLTYPE。

4. 逻辑备用数据库跳过的 SQL 语句。

CREATE/ALTER DATABASE

CREATE/ALTER/DROP MATERIALIZED VIEW

CREATE/ALTER/DROP MATERIALIZED VIEW LOG

CREATE/DROP DATABASE LINK

CREATE PFILE FROM SPFILE 和 CREATE SPFEIL FROM PFILE

CREATE SCHEMA AUTHORIZATION

ALTER SYSTEM、ALTER SESSION

CREATE CONTROL FILE

EXPLAIN

```
LOCK TABLE
SET CONSTRAINTS
SET ROLE
SET TRANSACTION
```

5. 确定逻辑备用数据库所支持的数据库方案。

```
SQL> select username from dba_users where username not in (select owner from dba_logstdby_skip);

USERNAME
------------------------------
SCOTT
HR
TSMSYS
```

6. 确定逻辑备用数据库不支持的对象。

```
SQL> select * from dba_logstdby_unsupported;

未选定行
```

7. 确保表行可以被唯一标识。

```
SQL> select * from dba_logstdby_not_unique;

OWNER      TABLE_NAME B
------- ---------- -
TSMSYS    SRS$           Y
SCOTT     BONUS          N
SCOTT     SALGRADE       N
```

8. 逻辑备份具体操作

ORACLE 数据库有两类备份方法。第一类为物理备份，该方法实现数据库的完整恢复，但数据库必须运行在归挡模式下（业务数据库在非归挡模式下运行），且需要极大的外部存储设备，例如磁带库；第二类备份方式为逻辑备份，业务数据库采用此种方式，此方法不需要数据库运行在归挡模式下，不但备份简单，而且可以不需要外部存储设备。数据库逻辑备份方法 ORACLE 数据库的逻辑备份分为三种模式：表备份、用户备份和完全备份。 -

2. 备份方案 -

1.表模式 -

备份某个用户模式下指定的对象（表）。业务数据库通常采用这种备份方式。若备份到本地文件，使用如下命令： -

exp icdmain/icd rows=y indexes=n compress=n buffer=65536 -

feedback=100000 volsize=0 -

file=exp_icdmain_csd_yyyymmdd.dmp -

log=exp_icdmain_csd_yyyymmdd.log -

tables=icdmain.commoninformation,icdmain.serviceinfo,icdmain.dealinfo 若直接备份到磁带设备，使用如下命令： -

exp icdmain/icd rows=y indexes=n compress=n buffer=65536 -

feedback=100000 volsize=0 -

file=/dev/rmt0 -

log=exp_icdmain_csd_yyyymmdd.log -

tables=icdmain.commoninformation,icdmain.serviceinfo,icdmain.dealinfo 注：在磁盘空间允许的情况下，应先备份到本地服务器，然后再拷贝到磁带。出于速度方面的考虑，尽量不要直接备份到磁带设备。 -

2.用户模式 -

备份某个用户模式下的所有对象。业务数据库通常采用这种备份方式。 若备份到本地文件，使用如下命令： -

exp icdmain/icd owner=icdmain rows=y indexes=n compress=n buffer=65536 -

feedback=100000 volsize=0 -

file=exp_icdmain_yyyymmdd.dmp -

log=exp_icdmain_yyyymmdd.log 若直接备份到磁带设备，使用如下命令： -

exp icdmain/icd owner=icdmain rows=y indexes=n compress=n buffer=65536 -

feedback=100000 volsize=0 -

file=/dev/rmt0 -

log=exp_icdmain_yyyymmdd.log 注：如果磁盘有空间，建议备份到磁盘，然后再拷贝到磁带。如果数据库数据量较小，可采用这种办法备份。 -

3.完全模式 -

备份完整的数据库。业务数据库不采用这种备份方式。备份命令为： -

exp icdmain/icd rows=y indexes=n compress=n buffer=65536 -

feedback=100000 volsize=0 full=y -

file=exp_fulldb_yyyymmdd.dmp(磁带设备则为/dev/rmt0) -

log=exp_fulldb_yyyymmdd.log 对于数据库备份，建议采用增量备份，即只备份上一次备份以来更改的数据。增量备份命令： -

exp icdmain/icd rows=y indexes=n compress=n buffer=65536 -

feedback=100000 volsize=0 full=y inctype=incremental -

file=exp_fulldb_yyyymmdd.dmp(磁带设备则为/dev/rmt0) -

log=exp_fulldb_yyyymmdd.log 注：关于增量备份必须满足下列条件： -

1. 只对完整数据库备份有效，且第一次需要 full=y 参数，以后需要 inctype=incremental 参数。

2. 用户必须有 EXP_FULL_DATABASE 的系统角色。 -

3. 话务量较小时方可采用数据库备份。 -

4. 如果磁盘有空间，建议备份到磁盘，然后再备份到磁带。业务数据库备份方法及周期用 EXP 进行备份前，先在 SYS 用户下运行 CATEXP.SQL 文件（如果以前已运行该文件，则不要执行这个脚本）。 -

没有特殊说明，不允许在客户端执行备份命令。备份命令参照表模式下的备份命令。 -

从磁盘文件备份到磁带，如果首先备份到本地磁盘文件，则需要转储到磁带设备上。 -

1. 若需查看主机上配置的磁带设备，使用如下命令： -

lsdev -Cc tape -

显示的结果如下例所示： -

rmt0 Available 30-58-00-2,0 SCSI 4mm Tape Drive -

rmt1 Defined 30-58-00-0,0 SCSI 4mm Tape Drive -

标明 Available 的设备是可用的磁带设备。2. 若需查看磁带存储的内容，使用如下命令： -

tar -tvf /dev/rmt0 -

显示的结果如下例所示： -

-rw-r--r-- 300 400 8089600 Jan 11 14:33:57 2001 exp_icdmain_20010111.dmp -

如果显示类似如下内容，则表示该磁带存储的备份数据是从数据库直接备份到磁带上， -

而非从本地磁盘转储到磁带的备份文件，因此操作系统无法识别。 -

tar: 0511-193 An error occurred while reading from the media. -

There is an input or output error. -

或 tar: 0511-169 A directory checksum error on media; -267331077 not equal to 25626.3. 对于新磁带或无需保留现存数据的磁带，使用如下命令： -

tar -cvf /dev/rmt0 exp_icdmain_yyyymmdd.dmp 注：A. 该命令将无条件覆盖磁带上的现存数据。-

B. 文件名不允许包含路径信息，如：/backup/exp_icdmain_yyyymmdd.dmp。

4. 对于需要保留现存数据的磁带，使用如下命令： -

tar -rvf /dev/rmt0 exp_icdmain_yyyymmdd.dmp 注：该命令将文件 exp_icdmain_yyyymmdd.dmp 追加到磁带的末端，不会覆盖现存的数据。特别强调：如果备份时是从数据库直接备份到磁带上，则不可再向该磁带上追加复制任何其他文件，否则该备份数据失效。

-

5. 若需将转储到磁带上的备份文件复制到本地硬盘，使用如下命令： -

A. 将磁带上的全部文件复制到本地硬盘的当前目录 -

tar -xvf /dev/rmt0 -

B. 将磁带上的指定文件复制到本地硬盘的当前目录 -

tar -xvf /dev/rmt0 exp_icdmain_yyyymmdd.dmp 备份时间安排 -

由于备份时对系统 I/O 有较大影响，所以，建议在晚上 11 点以后进行备份工作。 -

业务数据库 Oracle 版本的恢复 -

恢复方案需根据备份方案确定。由于业务数据库采用表备份和用户备份相结合的方案，所以业务数据库的恢复需根据实际情况采用表恢复和用户恢复相结合的方案。 -

3. 恢复方案 -

数据库的逻辑恢复分为表恢复、用户恢复、完全恢复三种模式。 -

1.表模式 -

此方式将根据按照表模式备份的数据进行恢复。 -

A. 恢复备份数据的全部内容 -

若从本地文件恢复，使用如下命令： -

imp icdmain/icd fromuser=icdmain touser=icdmain rows=y indexes=n -

commit=y buffer=65536 feedback=100000 ignore=n volsize=0 -

file=exp_icdmain_csd_yyyymmdd.dmp -

log=imp_icdmain_csd_yyyymmdd.log 若从磁带设备恢复，使用如下命令： -

imp icdmain/icd fromuser=icdmain touser=icdmain rows=y indexes=n -

commit=y buffer=65536 feedback=100000 ignore=n volsize=0 file=/dev/rmt0 -

log=imp_icdmain_csd_yyyymmdd.logB. 恢复备份数据中的指定表 -

若从本地文件恢复，使用如下命令： -

imp icdmain/icd fromuser=icdmain touser=icdmain rows=y indexes=n -

commit=y buffer=65536 feedback=100000 ignore=n volsize=0 -

file=exp_icdmain_csd_yyyymmdd.dmp -

log=imp_icdmain_csd_yyyymmdd.log -

tables=commoninformation,serviceinfo 若从磁带设备恢复，使用如下命令： -

imp icdmain/icd fromuser=icdmain touser=icdmain rows=y indexes=n -

commit=y buffer=65536 feedback=100000 ignore=n volsize=0 -

file=/dev/rmt0 -

log=imp_icdmain_csd_yyyymmdd.log -

tables=commoninformation,serviceinfo2.用户模式 -

此方式将根据按照用户模式备份的数据进行恢复。 -

A. 恢复备份数据的全部内容 -

若从本地文件恢复，使用如下命令： -

imp icdmain/icd fromuser=icdmain touser=icdmain rows=y indexes=n -

commit=y buffer=65536 feedback=100000 ignore=n volsize=0 -

file=exp_icdmain_yy -

yymmdd.dmp -

log=imp_icdmain_yyyymmdd.log 若从磁带设备恢复，使用如下命令： -

imp icdmain/icd fromuser=icdmain touser=icdmain rows=y indexes=n -

commit=y buffer=65536 feedback=100000 ignore=n volsize=0 file=/dev/rmt0 -

log=imp_icdmain_yyyymmdd.logB. 恢复备份数据中的指定表 -

若从本地文件恢复，使用如下命令： -

imp icdmain/icd fromuser=icdmain touser=icdmain rows=y indexes=n -

commit=y buffer=65536 feedback=100000 ignore=n volsize=0 -

file=exp_icdmain_yyyymmdd.dmp -

log=imp_icdmain_yyyymmdd.log -

tables=commoninformation,serviceinfo 若从磁带设备恢复，使用如下命令： -

imp icdmain/icd fromuser=icdmain touser=icdmain rows=y indexes=n -

commit=y buffer=65536 feedback=100000 ignore=n volsize=0 file=/dev/rmt0 -

log=imp_icdmain_yyyymmdd.log -

tables=commoninformation,serviceinfo3.完全模式 -

如果备份方式为完全模式，采用下列恢复方法： -

若从本地文件恢复，使用如下命令： -

imp system/manager rows=y indexes=n commit=y buffer=65536 -

feedback=100000 ignore=y volsize=0 full=y -

file=exp_icdmain_yyyymmdd.dmp -

log=imp_icdmain_yyyymmdd.log -

若从磁带设备恢复，使用如下命令： -

imp system/manager rows=y indexes=n commit=y buffer=65536 -

feedback=100000 ignore=y volsize=0 full=y -

file=/dev/rmt0 -

log=imp_icdmain_yyyymmdd.log 参数说明 -

1. ignore 参数 -

Oracle 在恢复数据的过程中，当恢复某个表时，该表已经存在，就要根据 ignore 参数的设置来决定如何操作。若 ignore=y，Oracle 不执行 CREATE TABLE 语句，直接将数据插入到表中，如果插入的记录违背了约束条件，比如主键约束，则出错的记录不会插入，但合法的记录会添加到表中。 -

若 ignore=n，Oracle 不执行 CREATE TABLE 语句，同时也不会将数据插入到表中，而是忽略该表的错误，继续恢复下一个表。 -

2. indexes 参数 -

在恢复数据的过程中，若 indexes=n，则表上的索引不会被恢复，但是主键对应的唯一索引将无条件恢复，这是为了保证数据的完整性。 -

字符集转换 -

对于单字节字符集（例如 US7ASCII），恢复时，数据库自动转换为该会话的字符集（NLS_LANG 参数）；对于多字节字符集（例如 ZHS16CGB231280），恢复时，应尽量使字符集相同（避免转换），如果要转换，目标数据库的字符集应是输出数据库字符集的超集。 -

恢复方法 -

业务数据库采用表恢复方案。在用 IMP 进行恢复前，先在 SYS 用户下运行 CATEXP.SQL 文件（如果以前已运行该文件，则不要执行这个脚本），然后执行下列命令： -

IMP ICDMAIN/ICD FILE=文件名 LOG=LOG 文件名 ROWS=Y -

COMMIT=Y BUFFER=Y IGNORE=Y TABLES=表名注：要恢复的表名参照备份的表名 -

恢复是在原表基础上累加数据 -

没有特殊说明，不允许在客户端执行恢复命令 -

4. IMP 常见问题及解决方法

(1) 数据库对象已经存在

一般情况, 导入数据前应该彻底删除目标数据下的表, 序列, 函数/过程,触发器等;
数据库对象已经存在, 按缺省的 imp 参数, 则会导入失败
如果用了参数 ignore=y, 会把 exp 文件内的数据内容导入
如果表有唯一关键字的约束条件, 不合条件将不被导入
如果表没有唯一关键字的约束条件, 将引起记录重复

(2) 数据库对象有主外键约束

不符合主外键约束时, 数据会导入失败
解决办法: 先导入主表, 再导入依存表
disable 目标导入对象的主外键约束, 导入数据后, 再 enable 它们

(3) 权限不够

如果要把 A 用户的数据导入 B 用户下, A 用户需要有 imp_full_database 权限

(4) 导入大表(大于 80M) 时，存储分配失败

默认的 EXP 时, compress = Y, 也就是把所有的数据压缩在一个数据块上.

导入时，如果不存在连续一个大数据块，则会导入失败.

导出 80M 以上的大表时，记得 compress= N, 则不会引起这种错误.

(5) imp 和 exp 使用的字符集不同

如果字符集不同，导入会失败，可以改变 unix 环境变量或者 NT 注册表里 NLS_LANG 相关信息.

导入完成后再改回来.

(6) imp 和 exp 版本不能往上兼容

imp 可以成功导入低版本 exp 生成的文件，不能导入高版本 exp 生成的文件

5. 实践代码： -

oracle 创建表空间，创建用户 -

//创建临时表空间 -

```
create temporary tablespace test_temp -
tempfile '/u01/app/oracle/oradata/orcl/test_temp01.Dbf' -
size 32m -
autoextend on -
next 32m maxsize 2048m -
extent management local; -
```

//创建数据表空间 -

```
create tablespace test_data -
logging -
datafile '/u01/app/oracle/oradata/orcl/test_data01.dbf' -
size 32m -
```

autoextend on -

next 32m maxsize 2048m -

extent management local; -

//创建用户并指定表空间 -

create user username identified by password -

default tablespace test_data -

temporary tablespace test_temp; -

//给用户授予权限 -

grant connect,resource to username; -

先创建一个用户和表空间，用户名 david，密码 david.在这个表空间下创建一个表：tianle。随便插入些数据。代码如下： -

SQL> create tablespace test_data -

3 datafile '/u01/app/oracle/oradata/orcl/test_data01.dbf' -

4 size 5m; -

Tablespace created. -

SQL> create user david identified by david default tablespace test_data; -

SQL> grant connect,resource to david; -

SQL> conn david/david -

SQL> create table tianle(id number, -

2 content varchar2(100)); -

SQL> set wrap off -

SQL> column id format a20; -

SQL> column content format a50; -

6. 几种备份举例-

✓ **表模式备份**

[oracle@roy orcl]$ exp david/david rows=y indexes=n compress=n buffer=65536 file=exp_tianle_090101.dmp log=exp_tianle_090101.log tables=(tianle); -

用户模式备份： -

[oracle@roy orcl]$ exp david/david owner=david rows=y indexes=n compress=n buffer=65536 file=exp_david__090101.dmp log=exp_david_090101.log; -

✓ **完全模式备份**

[oracle@roy orcl]$ exp david/david rows=y indexes=n compress=n buffer=65536 full=y file=exp_fulldatabase_090101.dmp log=exp_fulldatabase_090101.log; -

✓ **表模式恢复**

[oracle@roy orcl]$ imp david/david fromuser=david touser=david rows=y indexes=n commit=y buffer=65536 file=exp_tianle_090101.dmp log=imp_tianle_090101.log tables=(tianle); -

✓ **用户模式恢复**

[oracle@roy orcl]$ imp david/david fromuser=david touser=david rows=y indexes=n commit=y buffer=65536 file=exp_tianle_090101.dmp log=exp_tianle_090101.log; -

✓ **全库模式恢复**

[oracle@roy orcl]$ imp david/david rows=y indexes=n commit=y full=y ignore=y buffer=65536 file=/tmp/exp_fulldatabase_090101.dmp log=/tmp/imp.log; -

7. 对 Oracle 备份与恢复 的补充说明

之前曾整理过一片文章，来说明 Oracle 的备份与恢复的。

今天又看到了一些知识，与上次说明的角度不一样。 所以整理下，算是对上篇的一个补充说明。

Oracle 备份分逻辑备份和物理备份。

8. 逻辑备份

逻辑备份就是 exp/imp， 10g 以后推出了数据泵(Data Pump/ expdp/impdp). 数据泵在效率上要比之前的 exp/imp 高那么几倍，直观的反应就是备份所花的时间少了。 还有就是 Data Pump 的功能要比 exp/imp 多一点。

Oracle 10g EXPDP 和 IMPDP 使用说明

Oracle Database 10g 引入了最新的数据泵(Data Dump)技术,使 DBA 或开发人员可以将数据库元数据(对象定义)和数据快速移动到另一个 oracle 数据库中.

数据泵导出导入(EXPDP 和 IMPDP)的作用
1,实现逻辑备份和逻辑恢复.
2,在数据库用户之间移动对象.
3,在数据库之间移动对象
4,实现表空间搬移.

数据泵导出导入与传统导出导入的区别
在 10g 之前,传统的导出和导入分别使用 EXP 工具和 IMP 工具,从 10g 开始,不仅保留了原有的 EXP 和 IMP 工具,还提供了数据泵导出导入工具 EXPDP 和 IMPDP.使用 EXPDP 和 IMPDP 时应该注意的事项;
EXP 和 IMP 是客户段工具程序,它们既可以在可以段使用,也可以在服务段使用.
EXPDP 和 IMPDP 是服务端的工具程序,他们只能在 ORACLE 服务端使用,不能在客户端使用
IMP 只适用于 EXP 导出文件,不适用于 EXPDP 导出文件;IMPDP 只适用于 EXPDP 导出文件,而不适用于 EXP 导出文件.

数据泵导出包括导出表,导出方案,导出表空间,导出数据库 4 种方式.
EXPDP 命令行选项
1. ATTACH
该选项用于在客户会话与已存在导出作用之间建立关联.语法如下
ATTACH=[schema_name.]job_name
Schema_name 用于指定方案名,job_name 用于指定导出作业名.注意,如果使用 ATTACH 选项,在命令行除了连接字符串和 ATTACH 选项外,不能指定任何其他选项,示例如下:

Expdp scott/tiger ATTACH=scott.export_job
2. CONTENT
该选项用于指定要导出的内容.默认值为 ALL
CONTENT={ALL | DATA_ONLY | METADATA_ONLY}
当设置 CONTENT 为 ALL 时,将导出对象定义及其所有数据.为 DATA_ONLY 时,只导出对象数据,为 METADATA_ONLY 时,只导出对象定义
Expdp scott/tiger DIRECTORY=dump DUMPFILE=a.dump
CONTENT=METADATA_ONLY
3. DIRECTORY
指定转储文件和日志文件所在的目录
DIRECTORY=directory_object
Directory_object 用于指定目录对象名称.需要注意,目录对象是使用 CREATE DIRECTORY 语句建立的对象,而不是 OS 目录
Expdp scott/tiger DIRECTORY=dump DUMPFILE=a.dump
先在对应的位置创建物理文件夹，如 D:\backup
建立目录:
create or replace directory backup as '/opt/oracle/utl_file'
SQL>CREATE DIRECTORY backup as ‘d:\backup’;
SQL>grant read,write on directory backup to SYSTEM;
查询创建了那些子目录:
SELECT * FROM dba_directories;
4. DUMPFILE
用于指定转储文件的名称,默认名称为 expdat.dmp
DUMPFILE=[directory_object:]file_name [,….]
Directory_object 用于指定目录对象名,file_name 用于指定转储文件名.需要注意,如果不指定 directory_object,导出工具会自动使用 DIRECTORY 选项指定的目录对象
Expdp scott/tiger DIRECTORY=dump1 DUMPFILE=dump2:a.dmp
5. ESTIMATE
指定估算被导出表所占用磁盘空间分方法.默认值是 BLOCKS
EXTIMATE={BLOCKS | STATISTICS}
设置为 BLOCKS 时,oracle 会按照目标对象所占用的数据块个数乘以数据块尺寸估算对象占用的空间,设置为 STATISTICS 时,根据最近统计值估算对象占用空间
Expdp scott/tiger TABLES=emp ESTIMATE=STATISTICS
DIRECTORY=dump DUMPFILE=a.dump
6. EXTIMATE_ONLY
指定是否只估算导出作业所占用的磁盘空间,默认值为 N
EXTIMATE_ONLY={Y | N}
设置为 Y 时,导出作用只估算对象所占用的磁盘空间,而不会执行导出作业,为 N 时,不仅估算对象所占用的磁盘空间,还会执行导出操作.
Expdp scott/tiger ESTIMATE_ONLY=y NOLOGFILE=y
7. EXCLUDE
该选项用于指定执行操作时释放要排除对象类型或相关对象
EXCLUDE=object_type[:name_clause] [,….]
Object_type 用于指定要排除的对象类型,name_clause 用于指定要排除的具体对象.EXCLUDE

和 INCLUDE 不能同时使用
Expdp scott/tiger DIRECTORY=dump DUMPFILE=a.dup EXCLUDE=VIEW
8. FILESIZE
指定导出文件的最大尺寸,默认为 0,(表示文件尺寸没有限制)
9. FLASHBACK_SCN
指定导出特定 SCN 时刻的表数据
FLASHBACK_SCN=scn_value
Scn_value 用于标识 SCN 值.FLASHBACK_SCN 和 FLASHBACK_TIME 不能同时使用
Expdp scott/tiger DIRECTORY=dump DUMPFILE=a.dmp
FLASHBACK_SCN=358523
10. FLASHBACK_TIME
指定导出特定时间点的表数据
FLASHBACK_TIME=”TO_TIMESTAMP(time_value)”
Expdp scott/tiger DIRECTORY=dump DUMPFILE=a.dmp FLASHBACK_TIME=
“TO_TIMESTAMP(’25-08-2004 14:35:00’,’DD-MM-YYYY HH24:MI:SS’)”
11. FULL
指定数据库模式导出,默认为 N
FULL={Y | N}
为 Y 时,标识执行数据库导出.
12. HELP
指定是否显示 EXPDP 命令行选项的帮助信息,默认为 N
当设置为 Y 时,会显示导出选项的帮助信息.
Expdp help=y
13. INCLUDE
指定导出时要包含的对象类型及相关对象
INCLUDE = object_type[:name_clause] [,…]
14. JOB_NAME
指定要导出作用的名称,默认为 SYS_XXX
JOB_NAME=jobname_string
15. LOGFILE
指定导出日志文件文件的名称,默认名称为 export.log
LOGFILE=[directory_object:]file_name
Directory_object 用于指定目录对象名称,file_name 用于指定导出日志文件名.如果不指定 directory_object.导出作用会自动使用 DIRECTORY 的相应选项值.
Expdp scott/tiger DIRECTORY=dump DUMPFILE=a.dmp logfile=a.log
16. NETWORK_LINK
指定数据库链名,如果要将远程数据库对象导出到本地例程的转储文件中,必须设置该选项.
17. NOLOGFILE
该选项用于指定禁止生成导出日志文件,默认值为 N.
18. PARALLEL
指定执行导出操作的并行进程个数,默认值为 1
19. PARFILE
指定导出参数文件的名称
PARFILE=[directory_path] file_name

20. QUERY
用于指定过滤导出数据的 where 条件
QUERY=[schema.] [table_name:] query_clause
Schema 用于指定方案名,table_name 用于指定表名,query_clause 用于指定条件限制子句 .QUERY 选项不能与 CONNECT=METADATA_ONLY,EXTIMATE_ONLY,TRANSPORT_TABLESPACES 等选项同时使用.
Expdp scott/tiger directory=dump dumpfiel=a.dmp
Tables=emp query='WHERE deptno=20'
21. SCHEMAS
该方案用于指定执行方案模式导出,默认为当前用户方案.
22. STATUS
指定显示导出作用进程的详细状态,默认值为 0
23. TABLES
指定表模式导出
TABLES=[schema_name.]table_name[:partition_name][,…]
Schema_name 用于指定方案名,table_name 用于指定导出的表名,partition_name 用于指定要导出的分区名.
24. TABLESPACES
指定要导出表空间列表
25. TRANSPORT_FULL_CHECK
该选项用于指定被搬移表空间和未搬移表空间关联关系的检查方式,默认为 N.
当设置为 Y 时,导出作用会检查表空间直接的完整关联关系,如果表空间所在表空间或其索引所在的表空间只有一个表空间被搬移,将显示错误信息.当设置为 N 时,导出作用只检查单端依赖,如果搬移索引所在表空间,但未搬移表所在表空间,将显示出错信息,如果搬移表所在表空间,未搬移索引所在表空间,则不会显示错误信息.
26. TRANSPORT_TABLESPACES
指定执行表空间模式导出
27. VERSION
指定被导出对象的数据库版本,默认值为 COMPATIBLE.
VERSION={COMPATIBLE | LATEST | version_string}
为 COMPATIBLE 时,会根据初始化参数 COMPATIBLE 生成对象元数据;为 LATEST 时,会根据数据库的实际版本生成对象元数据.version_string 用于指定数据库版本字符串.

调用 EXPDP
使用 EXPDP 工具时,其转储文件只能被存放在 DIRECTORY 对象对应的 OS 目录中,而不能直接指定转储文件所在的 OS 目录.因此,使用 EXPDP 工具时,必须首先建立 DIRECTORY 对象.并且需要为数据库用户授予使用 DIRECTORY 对象权限.
CREATE DIRECTORY dump dir AS 'D:DUMP';
GRANT READ, WIRTE ON DIRECTORY dump_dir TO scott;
1,导出表
Expdp scott/tiger DIRECTORY=dump_dir DUMPFILE=tab.dmp TABLES=dept,emp logfile=exp.log;

2,导出方案
Expdp scott/tiger DIRECTORY=dump_dir DUMPFILE=schema.dmp SCHEMAS=system,scott logfile=/exp.log;

3.导出表空间
Expdp system/manager DIRECTORY=dump_dir DUMPFILE=tablespace.dmp TABLESPACES=user01,user02 logfile=/exp.log;

4,导出数据库
Expdp system/manager DIRECTORY=dump_dir DUMPFILE=full.dmp FULL=Y logfile=/exp.log;

使用 IMPDP
IMPDP 命令行选项与 EXPDP 有很多相同的,不同的有:
1,REMAP_DATAFILE
该选项用于将源数据文件名转变为目标数据文件名,在不同平台之间搬移表空间时可能需要该选项.
REMAP_DATAFIEL=source_datafie:target_datafile
2,REMAP_SCHEMA
该选项用于将源方案的所有对象装载到目标方案中.
REMAP_SCHEMA=source_schema:target_schema
3,REMAP_TABLESPACE
将源表空间的所有对象导入到目标表空间中
REMAP_TABLESPACE=source_tablespace:target_tablespace
4.REUSE_DATAFILES
该选项指定建立表空间时是否覆盖已存在的数据文件.默认为 N
REUSE_DATAFIELS={Y | N}
5.SKIP_UNUSABLE_INDEXES
指定导入是是否跳过不可使用的索引,默认为 N
6,SQLFILE
指定将导入要指定的索引 DDL 操作写入到 SQL 脚本中
SQLFILE=[directory_object:]file_name
Impdp scott/tiger DIRECTORY=dump DUMPFILE=tab.dmp SQLFILE=a.sql
7.STREAMS_CONFIGURATION
指定是否导入流元数据(Stream Matadata),默认值为 Y.
8,TABLE_EXISTS_ACTION
该选项用于指定当表已经存在时导入作业要执行的操作,默认为 SKIP
TABBLE_EXISTS_ACTION={SKIP | APPEND | TRUNCATE | FRPLACE }
当设置该选项为 SKIP 时,导入作业会跳过已存在表处理下一个对象;当设置为 APPEND 时,会追加数据,为 TRUNCATE 时,导入作业会截断表,然后为其追加新数据;当设置为 REPLACE 时,导入作业会删除已存在表,重建表病追加数据,注意,TRUNCATE 选项不适用与簇表和 NETWORK_LINK 选项
9.TRANSFORM
该选项用于指定是否修改建立对象的 DDL 语句

TRANSFORM=transform_name:value[:object_type]
Transform_name 用于指定转换名,其中 SEGMENT_ATTRIBUTES 用于标识段属性(物理属性,存储属性,表空间,日志等信息),STORAGE 用于标识段存储属性,VALUE 用于指定是否包含段属性或段存储属性,object_type 用于指定对象类型.
Impdp scott/tiger directory=dump dumpfile=tab.dmp Transform=segment_attributes:n:table
10.TRANSPORT_DATAFILES
该选项用于指定搬移空间时要被导入到目标数据库的数据文件
TRANSPORT_DATAFILE=datafile_name
Datafile_name 用于指定被复制到目标数据库的数据文件
Impdp system/manager DIRECTORY=dump DUMPFILE=tts.dmp
TRANSPORT_DATAFILES='/user01/data/tbs1.f'

调用 IMPDP
1, 导入表
Impdp scott/tiger DIRECTORY=dump_dir DUMPFILE=tab.dmp TABLES=dept,emp logfile=/exp.log;

Impdp system/manage DIRECTORY=dump_dir DUMPFILE=tab.dmp
TABLES=scott.dept,scott.emp REMAP_SCHEMA=SCOTT:SYSTEM logfile=/exp.log;

第一种方法表示将 DEPT 和 EMP 表导入到 SCOTT 方案中,第二种方法表示将 DEPT 和 EMP 表导入的 SYSTEM 方案中.
注意,如果要将表导入到其他方案中,必须指定 REMAP SCHEMA 选项.
2,导入方案
Impdp scott/tiger DIRECTORY=dump_dir DUMPFILE=schema.dmp SCHEMAS=scott logfile=/exp.log;

Impdp system/manager DIRECTORY=dump_dir DUMPFILE=schema.dmp
SCHEMAS=scott REMAP_SCHEMA=scott:system logfile=/exp.log;

3,导入表空间
Impdp system/manager DIRECTORY=dump_dir DUMPFILE=tablespace.dmp TABLESPACES=user01 logfile=/exp.log;

4,导入数据库
Impdp system/manager DIRECTORY=dump_dir DUMPFILE=full.dmp FULL=y logfile=/exp.log;

9. 物理备份

先说明一下，RMAN 属于物理备份。 Oracle 分为 Archive log 模式 和 NoArchive Log 模式。

1. NoArchive Log 模式下的物理备份

这种模式下只能在数据库关闭的情况下才能备份。 并且只能完全恢复到备份的时间点。

手工备份的步骤如下：

（1） 完全关闭数据库

（2） 备份所有数据库的数据文件，控制文件和联机重做日志

（3） 重新启动数据库

2. NoArchiveLog 模式下的恢复

很简单，关闭数据库后，把备份的数据文件，控制文件和联机重做日志还原，在启动数据库即可。 不过这种备份只能恢复到最后的备份时间点。

3. Archive Log 模式下的物理备份

3.1 脱机备份（冷备份）

步骤如下：

（1） 完全关闭数据库

（2） 备份所有数据库的数据文件

（3） 重新启动数据库

（4） 使用 alter system switch logfile 命令强制执行一个联机重做日志切换，一旦归档了联机重做日志，那么就备份所有的归档重做日志。

（5） 使用 alter database backup control file to trace 和 alter database backup control file to 'file_name' 命令创建控制文件的一个备份

3.2 联机备份（热备份）

步骤如下：

（1） 使用 alter tablespace ts_name begin backup 命令将需要备份的表空间和数据文件置入联机备份模式。

如果希望备份整个数据库，可以使用 alter database begin backup 命令。

（2） 备份与刚才置入备份模式的表空间相关的数据文件

（3） 执行 alter tablespace ts_name end backup 或者 alter database end backup，将表空间或者数据库从热备份模式中取出。

（4） 使用 alter system switch logfile 命令，强制执行一次 redo log 切换。 一旦完成切换，就备份所有的归档的重做日志。

说明：为什么要强制切换 redo log。因为恢复操作必须应用在备份期间生成的所有重做上。Oracle 在联机备份期间不断地物理更新数据文件（除了数据文件头）时，在备份操作期间存在数据块分离的可能性，这种可能性就会导致备份数据文件不一致。 此外，数据文件可能在备份之后，但是在整个备份进程结束之前被写入，由于备份中的每个数据文件可能会有不同的 SCN，因此数据文件备份映像会不一致，所以拥有在备份期间生成的重做以应用于恢复是非常重要的。

10. 联机备份的不足：

执行 Alter tablespace ts_name begin backup 和 alter database begin backup 命令时，重做日志会发生更改。一般来说，Oracle 只将更改矢量存储为重做记录，这些小型的记录指定义已经发生的更改。当数据文件处于联机备份模式时，Oracle 会记录数据文件的整块更改，而不是仅仅记录更改矢量，这意味着联机备份期间总的重做日志会大量增加，这样在热备份进程执行期间所需要的磁盘空间和 CPU 开销会收到影响。

而 RMAN 提供了不将表空间置入热备份模式而执行热备份的功能。 所以就避免了使用 I/0 操作，当结束数据文件的联机备份状态时，就可以进行正常的操作。

注意：

在 Archivelog 模式下，联机备份和脱机备份，不备份联机重做日志 和 控制文件，只是备份归档文件及创建备份的控制文件，这是因为在恢复期间不希望冒险重写联机重做日志或者控制文件。

为什么不恢复联机重做日志，因为在 Archive log 模式的恢复期间，联机重做日志中可能有最新的重做，这样当前的联机重做日志将被用于完全的时间点恢复。如果丢失了联机重做日志，就必须用所有的归档日志来执行时间点恢复。

不备份控制文件的原因和重做一样，因为当前的控制文件含有最新的联机重做日志信息和归档的重做日志信息。

4. Archive Log 模式下的恢复

4.1 Archive log 模式下的完全恢复

完全恢复，就是没有数据丢失的恢复，它的前提是所有联机重做日志文件都没有损坏，所有的归档文件都在。 如果丢失了 redo log 或者 归档文件，就需要执行时间点的恢复。如果丢失了控制文件的备份，就需要恢复控制文件并执行不完全恢复。

完全恢复的步骤如下：

（1）从备份中还原所有数据文件

（2）还原所有备份的归档

（3）加载数据库（startup mount）

（4）恢复数据库（recover database）

（5）Oracle 会提示应用归档重做日志，在提示符下输入：AUTO，Oracle 会自动应用所有重做日志。

（6）应用结束有，打开数据库。

补充：如果用 Recover database until cancel 命令会恢复所需的重做日志，在应用完最后一个归档的冲过日志后，需要输入 cancel 命令来结束日志应用。

4.2 Archive Log 模式下的表空间恢复和数据文件恢复

可以在数据库加载或者数据库打开时执行表空间或者数据文件的恢复。

表空间的恢复步骤如下：

（1） 是表空间脱机（alter tablespace tbname offilne）

（2） 还原与要恢复的表空间相关联的所有数据文件

（3） 联机恢复表空间（recover tablespace）

（4） 恢复完成后，使表空间联机（alter tablespace tnname online）

数据文件的恢复，它的有点是可以保持表空间的联机，步骤如下：

（1） 使数据文件脱机（alter database datafile 'file_name' offline）

（2） 还原所有要恢复的数据文件

（3） 联机恢复数据文件（recover datafile）

（4） 完成恢复后，使数据文件联机（alter database 'file_name' online）.

4.3 Archive Log 模式下时间点恢复

4.3.1 恢复到某一时间点数据库的步骤

（1）从备份中恢复所有数据文件，这个备份要在恢复时间点之前结束

（2）使用命令恢复数据库并应用 redo： recover database until time '2010-6-4'

（3）完成恢复后打开数据库

4.3.2 使用 SCN 号来恢复，步骤如下：

（1）从备份中恢复所有数据文件，这个备份要在恢复时间点之前结束

（2）使用命令恢复数据库并应用 redo： recover database until change '1287299'

（3）打开数据库

SCN 小知识补充：

查看 SCN:
SELECT dbms_flashback.get_system_change_number FROM dual;
SELECT CURRENT_SCN FROM V$DATABASE;

查看 SCN 和 timestamp 之间的对应关系：

select scn,to_char(time_dp,'yyyy-mm-dd hh24:mi:ss')from sys.smon_scn_time;

系统时间标记与 scn 的每 5 分钟匹配一次。举个例子，比如 scn:339988,339989 分别匹配 09-05-3013:52:00 和 09-05-30-13:57:00，则当你通过 as of timestamp 查询 09-05-30 13:52:00 或 09-05-30 13:56:59 这段时间点内的时间时，oracle 都会将其匹配为 scn:339988 到 undo 表空间中查找，也就说在这个时间内，不管你指定的时间点是什么，查询返回的都将是 09-05-30 13:52:00 这个时刻的数据。

21. Oracle 数据泵备份

1. 举例

数据泵是 Oracle 10g 的新特性，10g 以后的版本才有。关于数据泵的理论知识参考我的 Blog：

Oracle 10g EXPDP 和 IMPDP 使用说明

http://blog.csdn.net/tianlesoftware/archive/2009/10/16/4674224.aspx

Logicalbackup.sh

```
#!/bin/ksh
# ######################################################################
#
#                    created by tianlesoftware
#                          2010-7-7
#                    Email: tianlesoftware@vip.qq.com
# ######################################################################
# Oracle Environment settings
PATH=/usr/bin:/usr/ucb:/etc:.:/usr/X/bin:/bin
export PATH
ORACLE_SID=SID; pw=oracle ; export pw ; export ORACLE_SID
ORACLE_BASE=/dba/oracle; export ORACLE_BASE
ORACLE_HOME=$ORACLE_BASE/product/10.2.0/db_1;    export ORACLE_HOME
ORA_NLS33=$ORACLE_HOME/ocommon/nls/admin/data ; export ORA_NLS33
TNS_ADMIN=$ORACLE_HOME/network/admin ; export TNS_ADMIN
LD_LIBRARY_PATH=$ORACLE_HOME/lib ; export LD_LIBRARY_PATH
```

```
PATH=$ORACLE_HOME/bin:/usr/local/bin:/usr/ccs/bin:$PATH:/usr/sbin

export PATH

CLASSPATH=$ORACLE_HOME/jlib:$ORACLE_HOME/product/jlib ; export CLASSPATH

backup_dir=/u03/oradata/dump_backup_dir

#数据泵需要 directory，这里指定自己的目录就可以了
#SQL>CREATE DIRECTORY backup as '/u03/oradata/dump_backup_dir';
#SQL>grant read,write on directory backup to SYSTEM;

DMP_FILE=SID_`date +%d%m%Y_%H%M%S`.dmp

LOG_FILE=SID_`date +%d%m%Y_%H%M%S`.log
#

#    Let's start with an export of the database

#

expdp user/pwd DIRECTORY=dump_backup_dir DUMPFILE=$DMP_FILE
TABLESPACES=TS_NAMEE logfile=$LOG_FILE    parallel=3;

# 这里的 2 个说明，用户名和密码换成自己的，我这里是备份表空间。

# parallel 这个参数是控制并行度的，默认是 1，但对于数据库比较大的时候，可以设置 parallel，
这样可以较少备份的

#时间，但是设置并行会耗 CPU 资源，如果 CPU 资源比较紧张的话，就不要设了。

#

# Just to be safe (with space), we'll compress the export file

# 压缩 dmp 文件，较少对空间的占用

#

cd $backup_dir

compress *.dmp
```

```
#

# Let's delete the backups and logs that are more than 1 days old

# 删除超过一天的 dmp 文件

#

find $backup_dir -name "NewccsTablespace*" -mtime +1 -exec rm {} \;

# That's all
```

部署的时候，只要把这个 sh 脚本添加到 crontab 中就可以了。

添加方法： crontab -e 进入编辑状态，然后把脚本的路径和相关执行时间写上。保存后用 crontab -l 就可以查看内容：

```
$ crontab -l

45 1 * * * /u03/scripts/logicbackup.sh
```

前面 5 个参数的意义：

0～59 表示分

1～23 表示小时

1～31 表示日

1～12 表示月份

0～6 表示星期（其中 0 表示星期日）

2. Unix crontab 命令详解

crontab 命令的功能是在一定的时间间隔调度一些命令的执行。在/etc 目录下有一个 crontab 文件，这里存放有系统运行的一些调度程序。每个用户可以建立自己的调度 crontab。 - crontab 命令有三种形式的命令行结构： -

crontab [-u user] [file] -

crontab [-u user] [-e|-l|-r] -

crontab -l -u [-e|-l|-r] 第一个命令行中，file 是命令文件的名字。如果在命令行中指定了这个文件，那么执行 crontab 命令，则将这个文件拷贝到 crontabs 目录下；如果在命令行中没有制定这个文件，crontab 命令将接受标准输入（键盘）上键入的命令，并将他们也存放在 crontab 目录下。 -

命令行中-r 选项的作用是从/usr/spool/cron/crontabs 目录下删除用户定义的文件 crontab； -

命令行中-l 选项的作用是显示用户 crontab 文件的内容。 -

使用命令 crontab -u user -e 命令编辑用户 user 的 cron(c)作业。用户通过编辑文件来增加或修改任何作业请求。 -

执行命令 crontab -u user -r 即可删除当前用户的所有的 cron 作业。 -

作业与它们预定的时间储存在文件/usr/spool/cron/crontabs/username 里。username 使用户名，在相应的文件中存放着该用户所要运行的命令。命令执行的结果，无论是标准输出还是错误输出，都将以邮件形式发给用户。文件里的每一个请求必须包含以 spaces 和 tabs 分割的六个域。前五个字段可以取整数值，指定何时开始工作，第六个域是字符串，称为命令字段，其中包括了 crontab 调度执行的命令。 -

第一道第五个字段的整数取值范围及意义是： -

0～59 表示分 -

1～23 表示小时 -

1～31 表示日 -

1～12 表示月份 -

0～6 表示星期（其中 0 表示星期日） -

/usr/lib/cron/cron.allow 表示谁能使用 crontab 命令。如果它是一个空文件表明没有一个用户能安排作业。如果这个文件不存在，而有另外一个文件/usr/lib/cron/cron.deny,则只有不包括在这个文件中的用户才可以使用 crontab 命令。如果它是一个空文件表明任何用户都可安排作业。两个文件同时存在时 cron.allow 优先，如果都不存在，只有超级用户可以安排作业。

PROD10G-/data/oradata/BACKUP/PROD10G>crontab -l -

0,15,30,45 * * * * JAVA_HOME=/data/app/oracle/10.2.0/jdk /data/app/oracle/10.2.0/ccr/bin/emCCR -cron -silent start -

00 8,12,16 * * * /data/app/scripts/monitor/Ora_TableSpace_Mon.sh -

00 8,12,16 * * * /data/app/scripts/monitor/df.sh -

30 2 * * * /data/app/scripts/hotbackup/hot_database_backup.sh -

10 8,12,16 * * * /data/app/scripts/monitor/check_ind_unusable.sh -

10 8,12,16 * * * /data/app/scripts/monitor/check_maxfilesize.sh -

10 8,12,16 * * * /data/app/scripts/monitor/check_objectsize.sh -

可以对 job 进行修改. -

----------PS---------- -

EDITOR 环境变量设置: -

EDITOR=vi; -

export EDITOR -

或者 -

export EDITOR=vi; -

- linux crontab 时间格式：- 分 時 日 月 星期

43 21 * * * 21:43 执行
15 05 * * * 05:15 执行
0 17 * * * 17:00 执行
0 17 * * 1 每周一的 17:00 执行
0,10 17 * * 0,2,3 每周日,周二,周三的 17:00 和 17:10 执行
0-10 17 1 * * 每月 1 日从 17:00 到 7:10 每隔 1 分钟 执行
0 0 1,15 * 1 每月 1 日和 15 日和 一日的 0:00 执行
42 4 1 * * 每月 1 日的 4:42 分 执行
0 21 * * 1-6 周一到周六 21:00 执行
0,10,20,30,40,50 * * * * 每隔 10 分 执行
*/10 * * * * 每隔 10 分 执行
* 1 * * * 从 1:0 到 1:59 每隔 1 分钟 执行
0 1 * * * 1:00 执行
0 */1 * * * 每时 0 分 每隔 1 小时 执行
0 * * * * 每时 0 分 每隔 1 小时 执行
2 8-20/3 * * * 8:02,11:02,14:02,17:02,20:02 执行

30 5 1,15 * * 1 日 和 15 日的 5:30 执行

22. 权限控制

1. 授权 sysdba

Grant sysdba to u1;

2. 回收权限

Revoke sysdba from u1;

3. 如何知道有哪些管理员

Select * from V$PWFILE_USERS

4. 修改口令文件的最大用户数

Orapwd file=’/oracle/app/oracle/product/db_1/dbs/orapwdnbo’ pwassword=caojj123
Entries=10 force=y
这里指的是超级用户数，p a s s w d 是超级用户密码

5. 忘记 sys 密码咋办

重建 s y s 口令文件
Orapwd file=’xxx/xxxx/orapwdnbo’ password=wu emtries=5 force=y
注意：口令文件名=orapwd + instance_name

6. 授权 sysdba 时提示错误：password file is null

原因： 管理用户的个数已满

7. 进行 sysdba 授权时，报告错误：password file cannot be updated in shared mode

原因:参数文件 init+instance_name.ora 的 REMOTE_LOGIN_PASSWORDFILE 的值为 SHARED ，应该修改成：EXCLUSIVE 。 修改后需要重启数据库

创建口令文件时出错

提示错误：unable to open password-file

原因：REMOTE_LOGIN_PASSWORDFILE 的值被设成 NONE 了,应该修改成 EXCLUSIVE ；查看方法： show parameter remote

8. 如何查看当前登录 sql 终端

```
SELECT s.username "ORACLE USER"
,p.pid "PROC PID"
,s.process "SESS PID"
,s.osuser "OS User"
,s.terminal "Terminal"
,s.machine "Machine"
,s.sid "SESS ID"
,s.serial# "SESS Serial#"
FROM v$process p
,v$session s
WHERE p.addr = s.paddr
```

9. SQL 下执行操作系统命令

```
SQL> !ls
RMAN> host;                    //切换出来
        Ls
    Exit                       // 退出操作系统 terminal,返回 RMAN
```

10. 如何停止所有 sqlplus 窗口

```
$ killall sqlplus
```

或者 $ killall –i sqlplus (带 prompt 提示)

11. 查看登陆服务器的终端

Finger (or finger oracle) 或者 who

12. 服务器相关

修改服务器时间：Date –s “2010-07-26 10:47:19”

查看服务器运行时长： uptime

查看进程运行时长：ps -ef |egrep oracle| egrep mon | egrep -v egrep

强制要求修改 oracle 密码：chage -M 60 oracle (60 天强制要求 oracle 修改密码)

替换字符串： :%s/oldstr/newstr/g

回退编辑： :u

跨用户拷贝文件：scp –rp filename user@host:/home/user/filename 或 scp initRMUG.ora oracle@rmougdev2:.

如何在文件中搜索字符串：find. –name “*.sh” |xargs –I “sqlplus” 或者：find . -name "*.trc" -exec grep -i "error" '{}' \; -print

如何查看二进制文件中的字符串：strings users01.dbf | grep -i Denver

如何按大小排序检索文件： ls –alS 或 du -S .|sort -nr|head -5

如何查看目录大小：du -sh /home/oracle

如何清空文本文件：cat /dev/null > listener.log 或者 cp /dev/null > listener.log

文本文件行数、单词数、字符数统计： wc rmanback.bash （或 –l 行数 –w 单词数）

文件软链接：ln –s /home/oracle/script/xxx.sh /home/oracle/yyy.sh(xxx.sh 文件被软连接到 yyy.sh，硬盘中只有一份，可以为目录建立软连接)

Tar 命令：

文件打包：tar -cvf prodrel.tar *.sql

目录打包： tar –cvf temp.tar /home/oracle/scripts

打包并压缩：tar -cvf - /oracle/product/10.2 | gzip > orahome.tar.gz

解包：tar –xvf temp.tar 或指定文件提取：tar –xvf temp.tar ttt.sh

查看： tar -tvf orahome.tar 或 tar -tzvf orahome.tar.gz

添加文件到已有的归档文件：tar –rvf temp.tar xxx.sql(可以是目录)

Cpio 命令摘要：

打包：ls | cpio -ov > backup.cpio

目录打包：find . -depth | cpio -ov > orahome.cpio

解包：cpio -idvm < linux10g_disk1.cpio

解包 2：cat linux10g_disk1.cpio.gz | gunzip | cpio –idvm

查看：cpio -itv < orahome.tar 或 cpio -itv < dir.cpio

添加文件文件：ls *.sql | cpio -ovAF my.cpio

Zip 命令摘要：

zip -r ora10g.zip oracle10g

解压： unzip upgrade.zip upgrade.sql

查看：unzip -l backup.zip

天剑文件：zip -g my.zip script.sql

Gzip

压缩：gzip scripts.tar

解压：gunzip scripts.tar.gz

数据库物理迁移：

A、 查看数据迁数据库数据文件位置

SQL>select name from v$datafile

B、 拷贝数据文件

a) Cp … …

C、 删除当前数据库数据文件

a) Rm *

D、 创建数据库目录连接

a) Ln –s old new

E、 启动数据库

a) Startup

23. 服务运行情况分析

1. 常用分析工具介绍

- Vmstat
  监控系统资源使用情况：CPU、MEMERY
- Watch
  周期性的执行某种命令
- Ps
  进程信息查看命令
- Top
  进程资源查看命令
- Mpstat(没有该工具)
  CPU 性能统计
- Sar（没有该工具）
  显示查看 cpu、memery、i/o 使用情况，包括当前和过去的。
- Free
  查看空闲和已经使用的内存情况
- Df
  查看磁盘信息
- Du
  查看磁盘中文件大小信息
- Iostat
  显示 IO 使用情况
- Netstat
  监控网络状况

2. 识别系统瓶颈

通常情况下，通过 vmstat,来查看系统资源，识别系统瓶颈

- Vmstat 缺省输出

```
$ vmstat
procs -----------memory---------- ---swap-- -----io---- --system-- ----cpu----
 r  b   swpd   free   buff  cache   si   so    bi    bo   in   cs us sy id wa
14  0  52340  25272   3068 1662704    0    0    63    76    9   31 15  1 84  0
```

- **Vmstat 输出说明**

R：等待运行的处理器
B：表示进程睡眠数
Swpd：被使用的虚拟内存
Free：空闲的物理内存
Buff：缓存占用的内存数量
Cache：二级缓存占用内存数
Si：在磁盘存储器交换
So：内存交换磁盘
Bi：数据块读取速度
Bo：数据库写入速度
In：每秒系统中断数
Cs：每秒上下文切换速度
Us：用户级别的 cpu 总时间占用比率
Sy：系统级别的 cpu 占用时间比率
Id：cpu 空闲时间比率
Wa ： I/O 等待时间

- **Vmstat 使用**

.1 时间段统计

vmstat <interval in seconds> <number of intervals>

interval in seconds ：多少秒统计一次

number of intervals：总共统计多少次

.2 循环执行 vmstat

Watch –n 5 –d vmstat

.3 输出到文件

Vmstat 2 10 > vmstat.log

.4 制定 vmstat 数据单位

Vmstat –S m （显示字节数为兆）

- **Top 使用**

.1 基本使用

Top > txt.log

.2 指定用户监控

Top –U oracle

.3 指定时间间隔刷新

Top –d 5 // 五秒刷新一次

.4 指定 top 刷新次数

Top –n 2 // 刷新两次后，top 自动退出

.5 批量模式

Top –b

3. 数据库报表分析

SQL> @?/rdbms/admin/awrrpt.sql

```
SQL> @?/rdbms/admin/addmrpt.sql

SQL> @?/rdbms/admin/ashrpt.sql
```

24. 常用脚本

4. 查询 SQL 脚本

```
#!/bin/bash
ORACLE_SID=SCDEV
ORACLE_HOME=/orahome/oracle/product/10.2.0.1/db_1
PATH=$ORACLE_HOME/bin:$PATH
echo "select 'DB up' from dual;" | sqlplus -s system/foo
exit 0
```

5. ORACLE 进程检查

```
#!/bin/bash
SID=SAND
critProc=ora_smon
ps -ef | grep -v 'grep' | grep ${critProc}_$SID
if [ $? -eq 0 ]; then
echo "$SID is available."
else
echo "$SID has issues." | mail -s "problem with $SID" bbill@gmail.com
fi
exit 0
```

说明：$? 表示前一个命令执行结果， 0 表示成功。

6. 循环脚本

```
#!/bin/bash
SID_LIST="dev1 dev2 dev3"
critProc=ora_smon
for curSid in $SID_LIST
do
ps -ef | grep -v 'grep' | grep ${critProc}_$curSid
if [ $? -eq 0 ]; then
echo "$curSid is available."
else
echo "$curSid has issues." | mail -s "issue with $curSid" lellison@oracle.com
fi
```

```
done
```

7. 全库备份脚本

```
#!/bin/bash
ORACLE_SID=$1
rman target / <<EOF
backup database;
EOF
```

8. 特殊字符说明

```
$@        // 表示传入的参数列表
$#或$*    // 传入参数个数
$1 – $n   // 表示第几个传入参数
$?        // 最后命令的状态，为 0 表示命令执行成功
$0        // 表示当前脚本名称
$$        // 当前脚本进程 ID
$!        // 后台执行的进程个数
&&        // 表示在前一命令执行成功才继续
      grep ORA-00600 alert.log && echo "DB prob" | Mail -s "ORA 600 error" lare@orc.com
||        // 表示前一个命令执行不成功则继续
      ls alert.log || echo "no log" | Mail -s "log file error" larrye@oracle.com
${10}     // 访问传入的第十个参数
```