oracle 的入门心得（来自 selina33 版主）

oracle 的体系太庞大了，对于初学者来说，难免会有些无从下手的感觉，什么都想学，结果什么都学不好，所以把学习经验共享一下，希望让刚刚入门的人对 oracle 有一个总体的认识，少走一些弯路。

一、定位

oracle 分两大块，一块是开发，一块是管理。开发主要是写写存储过程、触发器什么的，还有就是用 Oracle 的 Develop 工具做 form。有点类似于程序员，需要有较强的逻辑思维和创造能力，个人觉得会比较辛苦，是青春饭 J；管理则需要对 oracle 数据库的原理有深刻的认识，有全局操纵的能力和紧密的思维，责任较大，因为一个小的失误就会 down 掉整个数据库，相对前者来说，后者更看重经验。

因为数据库管理的责任重大，很少公司愿意请一个刚刚接触 oracle 的人去管理数据库。对于刚刚毕业的年轻人来说，可以先选择做开发，有一定经验后转型，去做数据库的管理。当然，这个还是要看人个的实际情况来定。

二、学习方法

我的方法很简单，就是：看书、思考、写笔记、做实验、再思考、再写笔记

看完理论的东西，自己静下心来想想，多问自己几个为什么，然后把所学和所想的知识点做个笔记；在想不通或有疑问的时候，就做做实验，想想怎么会这样，同样的，把实验的结果记下来。思考和做实验是为了深入的了解这个知识点。而做笔记的过程，也是理清自己思路的过程。

学习的过程是使一个问题由模糊到清晰，再由清晰到模糊的过程。而每次的改变都代表着你又学到了一个新的知识点。

学习的过程也是从点到线，从线到网，从网到面的过程。当点变成线的时候，你会有总豁然开朗的感觉。当网到面的时候，你就是高手了

很多网友，特别是初学的人，一碰到问题就拿到论坛上来问，在问前，你有没有查过书，自己有没有研究过，有没有搜索一下论坛？这就叫思维惰性。由别人来回答你的问题，会让你在短时间内不费劲地弄懂这个知识点，然而通过自己的努力去研究它，不但会更深入的了解这个知识点，更重要的是在研究的过程会提高你解决问题和分析问题的能力。总的来说，没有钻研的学习态度，不管学什么东西，都不会成功的。

当然，初学的人很多时候是因为遇到问题时，无从下手，也不知道去哪里找资料，才会到论坛上

提问题的。但我认为，在提问的时候，是不是可以问别人是如何分析这个问题？从哪里可以找到相关的资料？而不是这个问题的答案是什么？授人以鱼不如授人以渔。

三、oracle 的体系

oracle 的体系很庞大，要学习它，首先要了解 oracle 的框架。在这里，简要的讲一下 oracle 的架构，让初学者对 oracle 有一个整体的认识。

1、物理结构（由控制文件、数据文件、重做日志文件、参数文件、归档文件、密码文件组成）
控制文件：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，一个数据库至少需要一个控制文件
数据文件：存储数据的文件
重做日志文件：含对数据库所做的更改记录，这样万一出现故障可以启用数据恢复。一个数据库至少需要两个重做日志文件
参数文件：定义 Oracle 例程的特性，例如它包含调整 SGA 中一些内存结构大小的参数
归档文件：是重做日志文件的脱机副本，这些副本可能对于从介质失败中进行恢复很必要。
密码文件：认证哪些用户有权限启动和关闭 Oracle 例程

2、逻辑结构（表空间、段、区、块）
表空间：是数据库中的基本逻辑结构，一系列数据文件的集合。
段：是对象在数据库中占用的空间
区：是为数据一次性预留的一个较大的存储空间
块：ORACLE 最基本的存储单位，在建立数据库的时候指定

3、内存分配（SGA 和 PGA）
SGA：是用于存储数据库信息的内存区，该信息为数据库进程所共享。它包含 Oracle 服务器的数据和控制信息，它是在 Oracle 服务器所驻留的计算机的实际内存中得以分配，如果实际内存不够再往虚拟内存中写。
PGA：包含单个服务器进程或单个后台进程的数据和控制信息，与几个进程共享的 SGA 正相反 PGA 是只被一个进程使用的区域，PGA 在创建进程时分配在终止进程时回收

4、后台进程（数据写进程、日志写进程、系统监控、进程监控、检查点进程、归档进程、服务进程、用户进程）
数据写进程：负责将更改的数据从数据库缓冲区高速缓存写入数据文件
日志写进程：将重做日志缓冲区中的更改写入在线重做日志文件
系统监控：检查数据库的一致性如有必要还会在数据库打开时启动数据库的恢复
进程监控：负责在一个 Oracle 进程失败时清理资源
检查点进程：负责在每当缓冲区高速缓存中的更改永久地记录在数据库中时,更新控制文件和数据文件中的数据库状态信息。
归档进程：在每次日志切换时把已满的日志组进行备份或归档
服务进程：用户进程服务。
用户进程：在客户端，负责将用户的 SQL 语句传递给服务进程，并从服务器段拿回查询数据。

5、oracle 例程：Oracle 例程由 SGA 内存结构和用于管理数据库的后台进程组成。例程一次只能打开和使用一个数据库。

6、SCN(System Change Number)：系统改变号，一个由系统内部维护的序列号。当系统需要更新的时候自动增加，他是系统中维持数据的一致性和顺序恢复的重要标志。

四、深入学习

管理：可以考 OCP 证书，对 oracle 先有一个系统的学习，然后看 Oracle Concepts、oracle online document,对 oracle 的原理会有更深入的了解，同时可以开始进行一些专题的研究如：RMAN、RAS、STATSPACT、DATAGUARD、TUNING、BACKUP&RECOVER 等等。

开发：对于想做 Oracle 开发的，在了解完 Oracle 基本的体系结构之后，可以重点关注 PL/SQL 及 Oracle 的开发工具这一部分。 PL/SQL 主要是包括怎么写 SQL 语句，怎么使用 Oracle 本身的函数，怎么写存储过程、存储函数、触发器等。 Oracle 的开发工具主要就是 Oracle 自己的 Developer Suite（Oracle Forms Developer and Reports Developer 这些），学会如何熟练使用这些工具。

五、开发经验及常见问题 （见附件）

常见 Oracle 安装问题说明 [当前帖子数：5]
1.问题集锦一
2.Oracle9i 在 RedhatLinux8.0 中的安装详细步骤
3.问题集锦二
4.如何配置和使用 iSQL*Plus
5.Oracle9i 中 Data Guard 的新特性以及配置使用

常见 Oracle 入门问题说明 [当前帖子数：5]
1.关于 Linux 下 DBSTART 和 DBSHUT 脚本中需要修改的地方
2.如何将 EXP 出来的数据 IMP 进不同的表空间
3.如果系统中安装了多个数据库实例，如何修改默认 SID
4.Oracle9i 初始化参数注解
5.关于 Oracle 数据库的升级(Migration)

常见 Oracle 开发问题说明 [当前帖子数：0]

Oracle 技巧与提示 [当前帖子数：3]
1.如何修改数据库的字符集

以下方法只适用于需要修改的字符集是当前字符集的超集，声明：以下方法请谨慎使用，同时一定要在修改字符集以前作当前数据库的完全冷备份。

[转载自 http://www.xray.mpe.mpg.de]
This article describes how one can change the character set of a database. It should be done with extreme caution having noted the following limitations.

The database character set should only be changed if the characters of the code page of the originating database correspond to the same characters of the target database, ie. if the database was created with the characterset US7ASCII and it is to be updated to WE8ISO8859P1. Since these have the same encoding scheme for the first 127 bits, changing the character set from US7ASCII to WE8ISO8859P1 will display all characters up to 127 bits as the same character before and after. In addition, in this particular example, if any characters have been entered with the 8th bit set, then updating the database characterset to 8 bit will allow that 8th bit to be displayed. You must not change the characterset from one encoding scheme to another encoding scheme where the code pages do not correspond. This will completely scramble your database. In addition, if case*designer diagrams are stored in the database, this method must not be used. Contact Worldwide Support for further details.

Before attempting to run any of the scripts below, you must take a full cold backup of your database. In addition, the procedure must be thoroughly tested before attempting this on a production instance.

Here's a SQL*PLUS script that allows a database's character set to be changed to a different encoding scheme without having to rebuild the database.

```
set echo off
set verify off

rem The data dictionary table that records the database
rem character set is sys.props$
rem
rem SQL> describe sys.props$
rem Name Null? Type
rem ------------------------------ -------- ----
rem NAME NOT NULL VARCHAR2(30)
rem VALUE$ VARCHAR2(2000)
rem COMMENT$ VARCHAR2(2000)

rem For example:
rem
rem SQL> column c1 format a30
rem SQL> select name c1, value$ c1 from sys.props$;
```

```
rem C1 C1
rem ---------------------------- ----------------------------
rem DICT.BASE 2
rem NLS_LANGUAGE AMERICAN
rem NLS_TERRITORY AMERICA
rem NLS_CURRENCY $
rem NLS_ISO_CURRENCY AMERICA
rem NLS_NUMERIC_CHARACTERS .,
rem NLS_DATE_FORMAT DD-MON-YY
rem NLS_DATE_LANGUAGE AMERICAN
rem NLS_CHARACTERSET WE8DEC
rem NLS_SORT BINARY
rem GLOBAL_DB_NAME NLSV7.WORLD

rem NLS_CHARACTERSET can be changed by updating its value, for example:

rem update sys.props$
rem set value$ = 'WE8ISO8859P1'
rem Where name = 'NLS_CHARACTERSET';

rem The database has to be shutdown and restarted before the change
rem becomes effective.

rem It is very important to specify the character set name correctly.
rem IMPORTANT NOTE
rem ==============
rem If NLS_CHARACTERSET is updated to an invalid value, it will not then
rem be possible to restart the database once it has been shutdown.
rem To recover, it will be necessary to re-create the database, since it
rem cannot be restarted to correct the invalid NLS_CHARACTERSET entry.

rem The character set name should be in uppercase.
rem The new value is not effective until the database has been shutdown and
rem restarted.
rem
rem A suggested procedure is as follows, and can be done by running this
rem script from SQL*Plus when logged into the SYSTEM account.
rem
rem USAGE : SQL> start ch_db.sql <character set>
rem
rem where <character set> is the desired database character set
rem
```

```
Prompt First check that the character set name is valid.

set echo on

select convert('a','&1','us7ascii') from dual;

set echo off

prompt If this select statement returns error ORA-01482, then the
prompt specified character set name is not valid for this installation.
prompt Abort the procedure now with Control-c

prompt To continue, press return
accept ans CHAR

Prompt Check the current value of database character set.

column c1 format a30
select name c1, value$ c1 from sys.props$
where name = 'NLS_CHARACTERSET';

prompt To continue, press return

Prompt Update to new character set

update sys.props$
set value$ = upper('&1')
where name = 'NLS_CHARACTERSET';

set echo off

prompt To continue, press return
accept ans CHAR

Prompt Check the new value of database character set

select name c1, value$ c1 from sys.props$
where name = 'NLS_CHARACTERSET';

Prompt If the value is updated as required, press return to continue and
Prompt then manually type COMMIT; to commit the change. Then shutdown and
Prompt restart the database.
Prompt
Prompt If the value is not updated as required, press return to continue and
```

Prompt than manually type ROLLBACK; to prevent the change.

prompt To continue, press return
accept ans CHAR
---------->

2.如何查看 Control File 中保存的内容

Control File 是二进制文件，用普通的方法很难知道其中到底保存了什么内容，但是 Oracle 却提供了一个 SQL*PLUS 命令来将 Control File 的内容 dump 到文本文件中。
方法如下：
以 SYSDBA 身份登入 SQL*PLUS
SQL> oradebug setmypid
SQL> oradebug dump controlf 3

将把 control file dump 到 USER_DUMP_DEST 初始化参数指定的目录下。
其中 3 为 dump level。
level 的解释如下：
1 ：only the file header
2 ：just the file header, the database info record, and checkpoint progress records
3 ：all record types, but just the earliest and latest records for circular reuse record types
4 ：as above, but includes the 4 most recent records for circular reuse record types
5+ ：as above, but the number of circular reuse records included doubles with each level

3.Oracle9i(Version 9.2)SYS_CONTEXT 函数的用法以及同 USERENV 函数的比较

六、共享

介绍几本 oracle 入门的好书
Oracle8i 备份恢复手册
Oracle8 高级管理与优化
Oracle8i PLSQL 程序设计
Oracle8 数据库管理员手册
Oracle9 statspack 高性能调整
以上书本都是机械工业出版社出版。
还有就是 OCP 的教学用书

介绍几个网站
http://tahiti.oracle.com oracle 的官方文档
http://metalink.oracle.com/ oracle 的技术支持网站。需要购买 Oracle 服务才能有一个帐号，才能登陆，有大量的 Knowledge Base，大量问题解决经验。

http://www.oracle.com oracle 的官方网站，可以在这里 down oracle 的软件、官方文档和获得最新的消息
http://www.dbazine.com/ Oracle 的杂志
http://asktom.oracle.com
http://www.orafaq.net/
http://www.ixora.com.au/
http://www.oracle-base.com
http://www.dba-oracle.com/oracle_links.htm

附件一：

一．常见 Oracle 安装问题说明（来自 Kamus 版主）

0。Oracle8i 在 P4 机器中的安装
http://www.itpub.net/showthread.php?s=&threadid=129726
http://www.itpub.net/showthread.php?s=&threadid=123748

1。Oracle 在 Windows98 中的安装
在 Windows98 或者 WindowsMe 中安装 Oracle 需要下载 Oracle Personal Edition
下载地址如下：
http://otn.oracle.com/software/prod...al/content.html
安装过程与其他的版本没有差异。

2。Oracle 9i 在 Red Hat Linux 7.3 下的安装
http://www.itpub.net/showthread.php?s=&threadid=40129

3。Oracle9i 在 RedhatLinux8.x 中的安装

Wang-apollo wang-apollo@21cn.com September, 2002
Edit by Seraphim seraphim@itpub.net 2002.11

一．前言
终于把 Oracle 9iR2 成功地装在了 RedHat 8.0 下。
网上的几篇文章对我安装系统 Oracle 9 有很大的帮助，非常感谢！我写此文，也是受他们的感召，希望能给大家提供帮助。

二．参考资料
有很多人不知道如何安装、管理和调整 Oracle，其实在 http://doc.oracle.com 下有大量 PDF 文件讲解 Oracle 的基本操作。当然，Oracle 9.2 在 Unix 操作系统（包括 AIX，HP，Linux，

Solaris 和 Tru64）的安装手册也包括在里面，文件名是 a96167.pdf。当然是 English，看不懂？那还搞什么 Oracle？！
但是，Oracle 提供的操作手册实在是太详细了，详细得我搞糊涂了：有太多 Requirements，Patches，Packages，Recommend 之类的。我已经替大家检查过了，把步骤都精减了，不必翻那本 200 多页的手册，按我的步骤就可以快速完成安装工作。
准备好了吗？Action！

三．配置各种参数
从我的几次安装经验来看，最重要的就是要正确配置参数，其次是安装好包文件。回想两次不成功的安装，问题都集中在我一时犯迷糊，设错了参数。
以下的操作除了特别声明外，都是以 root 身份进行的。

1．硬件配置要求
安装 Oracle 9i 要求的最低硬件配置如下：
Memory：512MB （我装的第二台是 256M 内存，也没有问题）
Swap space：1G
Disk space： Database software：3.5G；seed database：1G
Temporary disk space：400MB in /tmp

2．设置 Swap space
使用下列命令检查内存的大小：
grep MemTotal /proc/meminfo
使用下列命令检查交换空间的大小：
cat /proc/swaps
为达到 1G 的 Swap space 要求，创建临时交换空间，命令如下：
dd if=/dev/zero of=tmpswap bs=1k count=1000000
chmod 600 tmpswap
mkswap tmpswap（如果是以 oracle 用户 telnet 上 server，再 su 成 root 的，则可能会报错说没有这个命令，那是因为 PATH 中没有/sbin 目录，所以运行/sbin/ mkswap tmpswap 就可以了）
swapon tmpswap

注意：重新启动后，临时交换空间不会自动加载，需要再次执行 swapon tmpswap 命令。
安装完毕后，可以用 swapoff tmpswap; rm tmpswap 命令删除交换空间和临时文件。

3．设置共享内存
Oracle 9i 使用 Solaris 的共享内存、交换区等资源进行工作，因而内核参数的是 Oracle 9i 能否正常安装和运行的关键，我第一次安装失败就是因为共享内存设置不够大导致中途退出的。a96167.pdf 对共享内存的大小没有明确的要求，但实际上不设置此项，极可能导致安装到中途时，安装程序突然退出。建议将共享内存的大小设置为 1G。使用如下命令检查共享内存的大小：
cat /proc/sys/kernel/shmmax
使用如下命令检查共享内存为 1G：
echo 1073741824 > /proc/sys/kernel/shmmax
设置完毕后，可以再运行命令查看是否更改了。每次系统启动后，需要重新设置。

4．设置临时目录
如果 /tmp 目录下没有 400MB 的容量，可以设置临时目录：
mkdir /home/tmp
chmod 1777 /home/tmp
set TEMP=/home/tmp; export TEMP
set TMPDIR=/home/tmp; export TMPDIR
安装完毕后，键入命令：
unset TEMP; unset TMPDIR
即可取消临时目录。

5．检查包文件和补丁程序
在 a96167.pdf 文件中，有需要安装的包文件和补丁程序的详细清单，此处不再列出。比较重要的有：
c++、JRE、JDK 等。以 Redhat 7.1 为例，从光盘上将如下一些文件复制到某个目录：
cpp-2.96-81.i386.rpm
glibc-devel-2.2.2-10.i386.rpm
kernel-headers-2.4.2-2.3.i386.rpm
gcc-2.96-81.i386.rpm
binutils-2.10.91.0.2-3.i386.rpm
然后执行如下命令，将包文件安装到系统中：
rpm -ivh cpp-2.96-81.i386.rpm \
glibc-devel-2.2.2-10.i386.rpm \
kernel-headers-2.4.2-2.3.i386.rpm \
gcc-2.96-81.i386.rpm \
binutils-2.10.91.0.2-3.i386.rpm
如果你的其它程序需要某个版本的包文件，可以在 Oracle 9i 安装完毕后，重新释放原来的包文件。

强制更新包文件的命令如下：
rpm -Uvh --force 包文件

从 http://www.blackdown.org 或 http://java.sun.com 下载 Blackdown 1.1.8_v3 文件，用如下命令安装到系统中：
bzip2 -dc jdk118_v3-glibc-2.1.3.tar.bz2 | tar xf - -C /usr/local
ln -s /usr/local/jdk118_v3 /usr/local/java

可以用如下命令检查包文件的安装状况：
rpm -qa

6．设置用户和组
Oracle 在安装和使用中需要用特定用户，不能用 root 用户来安装和运行 Oracle 9i。推荐使用 oracle 帐号。另外，安装 Oracle HTTP Server 时，需要一个 apache 的管理员，推荐使用 apache 帐号。操作步骤如下：

groupadd dba
groupadd oinstall
useradd -d /export/home/oracle -g oinstall -G dba –m oracle
passwd oracle
Linux8.0 中已经有 apache 用户了，所以下面可以省略：
groupadd apchadm
useradd -d /export/home/apache -g oinstall -G apchadm –m apache
passwd apache

7．设置安装目录
oracle9i 的典型安装需要至少两个安装点：一个安装基本的运行程序；一个为存放数据库，为简便，我将它们放在了同一个目录下，分别是/u01/oracle 和/u01/oracle/product/9.2.0。
mkdir /u01/oracle
mkdir /u01/oracle/product
mkdir /u01/oracle/product/9.2.0
chown -R oracle:oinstall /u01/oracle

8．设置 oracle 用户的环境变量
用 vi 之类的文件编辑工具打开 oracle 用户的环境变量文件。我的设置如下：
# Oracle Environment
ORACLE_BASE=/u01/oracle; export ORACLE_BASE
ORACLE_HOME=/u01/oracle/product/9.2.0; export ORACLE_HOME
ORACLE_SID=oralinux; export ORACLE_SID
ORACLE_TERM=xterm; export ORACLE_TERM
ORACLE_OWNER=oracle; export ORACLE_OWNER
# TNS_ADMIN=/export/home/oracle/config/9.0.1; export TNS_ADMIN
NLS_LANG=AMERICAN_AMERICA.ZHS16GBK; export NLS_LANG
ORA_NLS33=$ORACLE_HOME/ocommon/nls/admin/data; export ORA_NLS33
LD_LIBRARY_PATH=$ORACLE_HOME/lib:/lib:/usr/lib:/usr/local/lib
LD_LIBRARY_PATH==$LD_LIBRARY_PATH:$ORACLE_HOME/jdk/jre/lib/i386
LD_LIBRARY_PATH==$LD_LIBRARY_PATH:$ORACLE_HOME/jdk/jre/lib/i386/server
LD_LIBRARY_PATH==$LD_LIBRARY_PATH:$ORACLE_HOME/rdbms/lib

export LD_LIBRARY_PATH

Set shell search paths:

# PATH=$PATH:$HOME/bin:$ORACLE_HOME/bin
PATH=$PATH:/usr/sbin:/etc:/sbin
PATH=$PATH:$ORACLE_HOME/bin:/usr/local/samba/bin
export PATH

#CLASSPATH must include the following JRE locations:

CLASSPATH=$ORACLE_HOME/JRE:$ORACLE_HOME/jlib:$ORACLE_HOME/rdbms/jlib
CLASSPATH=$CLASSPATH:$ORACLE_HOME/network/jlib

9．下载安装文件
可以从 http://otn.oracle.com/ 网站下载 Oracle 9i for Linux 的文件，不过你可要遵守它的 License，以免产生版权问题。
下载的 3 个文件如下：lnx_920_disk1.cpio.gz，lnx_920_disk2.cpio.gz，lnx_920_disk3.cpio.gz。

将其放到临时目录，例如/files，更改文件属主：
chown oracle.oinstall lnx_920_disk1.cpio.gz
chown oracle.oinstall lnx_920_disk2.cpio.gz
chown oracle.oinstall lnx_920_disk3.cpio.gz
将上述 3 个文件解压，命令如下：
zcat lnx_920_disk1.cpio.gz | cpio -idmv
zcat lnx_920_disk2.cpio.gz | cpio -idmv
zcat lnx_920_disk3.cpio.gz | cpio –idmv
解压后，在临时目录下形成 disk1，disk2，disk3 等 3 个目录。可以用这些目录安装 Oracle 9i。也可以将这 3 个目录刻到光盘上，用光盘安装。

五．安装 Oracle
在服务器端，以 oracle 身份登录：
如果用硬盘中的安装文件，运行安装命令：
cd /files （存放 Oracle 9i 安装文件的临时目录）
disk1/runInstaller
如果是用光盘安装，运行安装命令：
/mnt/cdrom/runInstaller
终于可以见到 Oracle 欢迎界面了，学名是 OUI 安装程序

六．测试
安装完毕后，可以进行测试了：
以 oracle 用户登录，键入命令：
sqlplus /nolog
SQL>connect / as sysdba
SQL>startup

键入命令，启动监听程序：
lsnrctl start

启动 Oracle Web Server
cd $ORACLE_HOME/Apache/Apache/bin
./startJServ.sh
/u01/oracle/product/9.2.0/Apache/Apache/bin/apachectl start: httpd started

启动 Oracle Web Server 后默认的端口号是 7777
在客户端浏览器地址栏输入 http://xxx.xx.xxx.xxx:7777/

七．如何删除不成功的安装
如果安装不成功，想重新安装一遍的话，除了运行安装程序 runInstaller，进入 OUI 界面，删除安装的程序外，还应当执行命令删除如下目录和文件，重新启动系统：
rm -rf /etc/oraInst.loc /etc/oratab /tmp/OraInstall
/tmp/<FilesOwnedbyOracle>
删除 $ORACLE_BASE 下的所有内容后，再重新创建 product/9.2.0 目录。
重新启动系统后，别忘了重新设 swap space 和 shared memory 啊！

4。Oracle9i 在 RedhatLinux9.0 中的安装
http://www.itpub.net/showthread.php?s=&threadid=113676

5。Oracle10ibeta 在 RedhatLinux9.0 中的安装
http://www.itpub.net/showthread.php?s=&threadid=131403
http://www.itpub.net/showthread.php?s=&threadid=132054

6。Oracle9i 在 AIX 上的安装
http://www.itpub.net/showthread.php?s=&threadid=135397

7。如何在字符终端的 UNIX 上安装 Oracle
http://www.itpub.net/showthread.php?s=&threadid=1568

8。完整的手工建立 ORACLE 数据库步骤
http://www.itpub.net/showthread.php?s=&threadid=2616
http://www.itpub.net/showthread.php...15&pagenumber=2

9。Standby Database 的解决方案
http://www.itpub.net/showthread.php?s=&threadid=131401
http://www.itpub.net/showthread.php?s=&threadid=14933
http://www.itpub.net/showthread.php?s=&threadid=7888

10。Windows2000 AD 上安装 oracle9.2.0 单机 RAC 的大致方法
http://www.itpub.net/showthread.php?s=&threadid=125420

11。配置透明网关的步骤
http://www.itpub.net/showthread.php?s=&threadid=88909

12。AIX 5L 上安装 RAC 的过程
http://www.itpub.net/showthread.php?s=&threadid=84942

13。Oracle，HA 在 Unix 上双机环境的安装指南
http://www.itpub.net/showthread.php?s=&threadid=82899

14。配置 Oracle Name Server 的完全步骤
http://www.itpub.net/showthread.php?s=&threadid=8954

15。如何改变数据库实例名称
http://www.itpub.net/showthread.php?s=&threadid=3765

16。如何配置和使用 iSQL*Plus

Written by Seraphim seraphim@itpub.net 2003.06

关于 iSQL*Plus 的其他讨论，见以下链接：
http://www.itpub.net/showthread.php?s=&threadid=137308

其实使用 iSQL*Plus 非常方便，几乎不用什么配置，但是因为 DBA 用户和普通用户使用的是不同的 URL，可能会造成一部分人的疑惑，所以在这里把关于 iSQL*Plus 的各方面再详细说一下。

1。确认安装了 Oracle HTTP Server 和 iSQL*Plus Server

2。在安装的结尾部分将会显示登陆 iSQL*Plus 的默认端口，如果没有看到或者忘记了，那么可以检查如下文件来确认：
NT：%ORACLE_HOME%\Apache\Apache\conf\httpd.conf
UNIX：$ORACLE_HOME/Apache/Apache/bin/conf/httpd.conf
查看 Port 和 Listen 的参数值，默认应该是：
Port 7778
Listen 7778（HTTP）
Listen 4443（HTTPS）

可以通过编辑 oracle_apache.conf 配置文件启用或禁用 iSQL*Plus，注释以下行即可禁用
include "ORACLE_HOME\sqlplus\admin\isqlplus.conf

3。iSQL*Plus 的配置文件在
NT：%ORACLE_HOME%\sqlplus\admin\isqlplus.conf
UNIX：$ORACLE_HOME/sqlplus/admin/isqlplus.conf

4。确认已经启动了 Oracle HTTP Server

5。登录
登入普通用户，输入以下 URL：
http://machine_name.doman:port/isqlplus
登入 DBA 用户，输入以下 URL：
http://machine_name.domain:port/isqlplusdba
获取 Server 的统计信息：

http://machine_name.domain:port/isqlplusdba?statistics={active|full}
[&refresh=number]
其中 refresh 的最小值是 10 秒

6。如果要登录使用 SYSDBA 或 SYSOPER 权限登录到 iSQL*Plus，以通过 iSQL*Plus 执行数据库管理并运行 DBA 命令，则必须要 Oracle HTTP Server 验证，这个用户名和密码跟数据库的用户名密码无关。当然通过验证之后，还需要输入 Oracle 数据库中的具有 SYSDBA 或者 SYSOPER 权限的用户名和密码。
要通过 Oracle HTTP Server 验证，则必须将用户名和口令添加到 Oracle HTTP Server 验证文件中（用户验证文件位于%ORACLE_HOME%\sqlplus\admin\iplusdba.pw），作如下操作：
a。进入%ORACLE_HOME%\Apache\Apache\bin 目录
b。运行 htpasswd 实用程序：
C:\oracle\ora92\Apache\Apache\bin>htpasswd -help
Usage:
htpasswd [-cmdps] passwordfile username
htpasswd -b[cmdps] passwordfile username password

htpasswd -n[mdps] username
htpasswd -nb[mdps] username password
-c Create a new file.
-n Don't update file; display results on stdout.
-m Force MD5 encryption of the password (default).
-d Force CRYPT encryption of the password.
-p Do not encrypt the password (plaintext).
-s Force SHA encryption of the password.
-b Use the password from the command line rather than prompting for it.
On Windows, TPF and NetWare systems the '-m' flag is used by default.
On all other systems, the '-p' flag will probably not work.

比如运行：
htpasswd C:\oracle\ora92\sqlplus\admin\iplusdba.pw dbmanager
Automatically using MD5 format on Windows.
New password: *********
Re-type new password: *********
Adding password for user dbmanager

其它相关：
lionhp 关于 9i HTTP Server 在 Linux 平台下不能正常使用的错误及相关解决办法的帖子：
http://www.itpub.net/showthread.php?s=&threadid=149641
http://www.itpub.net/showthread.php?s=&threadid=150253

17。如何配置和使用 Oracle9i 的 DataGuard

Written by Seraphim seraphim@itpub.net 2003.07

DataGuard 是 Oracle8i 中的 Standby Database 技术的升级，原来 Standby 中为人所诟病的几个缺点均已得到解决，先来看一下 DataGuard 的新特性：
1。StandbyDatabase 现在分成了两种类型，一种称为 Physical standby database，这个和 8i 的 Standby 区别不大，另一种称为 Logical standby database，这是激动人心的新特性所在，也是本文将测试的类型

2。明确了几个 Service，一是 Log Transport Services，用于传送 redo 数据，区分了几种 database protection modes；一是 Log Apply Services，负责按照不同的类型将 redo 数据恢复到 standby database 中，也正是这个 service 体现了 Physical 和 Logical 两种类型的不同；另外一个是 Role Management Services，用于 standby 和 primary 角色的互相转换（刚看文档，感觉这一点应该也是一个提高，因为在 8i 中 standby 转为 primary 之后就无法重新转换为 standby 了）

3。增加了 Data Guard Broker，这是一个对 DataGuard 系统进行集中管理的工具，同时提供了对于 standby 维护的图形化界面

4。当 standby 为 Logical standby database 类型时，数据在作 recover 的时候（确切地说，不应该是 recover，因为对于此类型的 standby，Log Apply Services 将把 redo 数据转化成 SQL，然后通过 SQL 的执行使 standby 和 primary 同步），允许查询。在表面上和效果上我们可以看成这是一个 managed recovery 和 read only 模式的综合，多么激动人心！

5。对于 gap 的自动探测和恢复，所谓 gap 就是由于一些原因比如网络暂时不通等等产生的 standby 和 primary 之间 redo 数据的差异，在 8i 中如果出现这样的 gap，则必须手动将 gap 的 redo log file 传送到 standby 中，然后手动 recovery，之后才能再次进入 managed recovery 模式。而在 9i 中已经可以自动监测并且自动传送和恢复了。这又是一个多么激动人心的提高！

新特性介绍完毕，下面开始配置以及测试。
--TO BE CONTINUED

17。如何配置和使用 Oracle9i 的 DataGuard（续）

刚刚测试完 Oracle9i Data Guard Logical Standby Database，没有使用图形界面，完全按照文档中所说以 SQL*PLUS 命令操作，上贴中的 4 和 5 均作过测试，确实很不错，而且另外一个感觉就是文档写的太详细了，一步一步下来几乎不会有什么问题，所以就不再对于操作步骤多罗嗦了。
data guard 的文档请从 OTN 上下载。
Oracle Data Guard Concepts and Administration Release 2 (9.2)
October 2002
Part No. A96653-02

安装前需要检查的地方：

在整个 Data Guard 配置中必须在所有的系统里都安装相同版本的 Oracle Enterprise Edition

Primary database 必须运行在 ARCHIVELOG 模式下

Primary 和 standby 必须运行相同发行版本的（same release）Oracle 数据库。操作系统也必须相同，但是操作系统的版本可以不同（也就是说必须同时是 Windows 或者同时是 Unix，但是可以 primary 是 WindowsNT4，而 standby 是 Windows2000 Server）

硬件和操作系统的架构必须一样，比如不能 primary 是 64-bit 而 standby 是 32-bit

Primary 和 standby 都可以是单一的 instance 或者是 RAC 的 multi-instance

硬件的配置可以不同，比如 primary 和 standby 的 CPU 数，内存大小已经 storage 参数的设定

每个 primary 和 standby 都必须具有自己的 control file

用以管理 primary 和 standby 的用户必须具有 SYSDBA 权限

以下说一下注意点：

1。文档中提到需要用 nid 实用程序来修改 standby database 的 SID，感觉这一步不是必须的，如果要修改的话，应该先修改，再创建 Windows 下的 Service，而文档中的顺序是先创建 service 后修改 SID，结果我改了之后，只好把原先创建的 Service 删除再重新创建。

2。如果是 Windows 操作系统，并且以前已经建立过数据库，那么注意需要修改注册表中的 SID，指定为此次 standby 的 SID

3。该文档中对于如何修改 tnsnames.ora 和 listener.ora 没有多做描述，可以参考 Oracle8i standby 的文档，或者 Oracle net config 的文档

二．常见 Oracle 入门问题说明（来自 Kamus 版主）

1．关于 Linux 下 DBSTART 和 DBSHUT 脚本中需要修改的地方

Written by Seraphim(Mail: seraphim@itpub.net) 2003-06

系统环境：Redhat Linux 9 + Oracle9.2.0.1
在 Linux 系统下安装完数据库之后，会在$ORACLE_HOME/bin 下生成 dbstart 和 dnshut 脚本，这两个脚本可以简便地实现启动和关闭数据库。
这两个脚本运行时会读取/etc/oratab 文件，在这个文件里指定了需要启动和关闭的 SID（相

应 SID 那行的最后一个字符是 Y 而不是 N）

DBSHUT 的问题：
默认是执行 shutdown 而不是 shutdown immediate，这样当有别的 client 连着的时候，数据库不会 shutdown，可以把该脚本执行 shutdown 的部分改成 shutdown immediate，当然是不是需要这样强行切断用户连接，rollback 所有未 commit 的 transaction，还需要看自己的需求了。

DBSTART 问题：
执行时会检查在$ORACLE_HOME/dbs 中有没有 initSID.ora 文件，如果没有则报错退出。但是安装 9i 的时候通常会使用 spfile，所以在此目录下是不会存在 initSID.ora 文件的。修改的方法有两个：
一是改脚本，在 else 后面加判是否存在 spfile，如果有继续，没有再报错，但是此方法比较麻烦
二是创建一个 pfile，用 create pfile=pfilepath from spfile=spfilepath 就可以了，此命令在数据库 instance 没有启动的情况下也可以执行。
感觉这是 Oracle 的一个遗留问题，因为可以看到即使是检查了 pfile，Oracle 的启动仍然使用了 spfile。

2．如何将 EXP 出来的数据 IMP 进不同的表空间

Written by Seraphim(Mail: seraphim@itpub.net) 2003-07

经常有人提问：原来的数据在 USERS 表空间里面，我想把它 IMP 进 APP 表空间，我已经修改了目的用户的默认表空间，为什么结果还是 IMP 到 USERS 表空间中了呢。

关于此问题，作如下解释：
Oracle 并没有提供什么参数来指定要导入哪个表空间，数据默认将导入到原本导出时数据所在的表空间中，但是我们可以通过以下的方法来实现导入到不同的表空间。

1．在 IMP 时候使用 INDEXFILE 参数
当给此参数指定了某一文件名，IMP 的时候所有的 index 将不会直接导入到表空间中，而是在指定的文件中生成创建 index 的脚本。然后用文本编辑器打开此文件，直接编辑脚本中的 storage 参数，修改为想要导入的表空间名称。然后进入 SQL*PLUS，直接运行此脚本。最后重新执行 IMP，使用 INDEXS=n 参数将其余的 Objects 导入。
该方法适用于将 index 以及 constraints 导入指定的表空间。

2．改变目的用户的默认表空间
这就是上面说的经常有人提问的方法。但是上述的问题之所以没有成功，是因为缺少了下面的几步。
首先，收回目的用户的"UNLIMITED TABLESPACE"权限，revoke unlimited tablespace from username。
其次，取消目的用户在原数据导出表空间中的配额，这样才能迫使 IMP 把数据导入到用户的默认表空间中去。

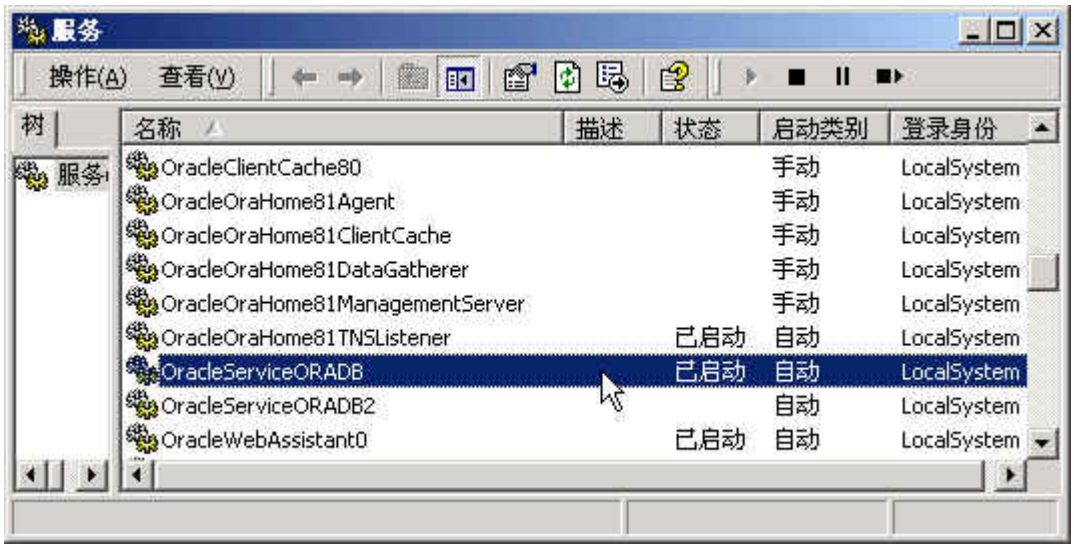
然后，将希望导入的表空间设为目的用户的默认表空间，并添加配额。
最后，执行 IMP。

3．如果系统中安装了多个数据库实例，如何修改默认 SID

此为 NT 系统中的设置
如果是在 UNIX 系统下，只需要简单地将 profile 中的 ORACLE_SID 改成希望的 SID 值即可。

- 系统环境：

  1、操作系统：Windows 2000
  2、数据库： Oracle 8i R2 (8.1.6) for NT 企业版
  3、安装路径：C:\ORACLE

- 设置方法：
  - 1、假设安装了两个数据库，分别为 oradb 和 oradb2
  - 

  - 先安装 oradb，再安装 oradb2 后，所有工具的默认连接库均为 oradb2
  - 2、regedit
  - 定位至：HKEY_LOCAL_MACHINE\SOFTWARE\ORACLE\HOME0
  - 找到 ORACLE_SID 这一项，将它的值改为 oradb 即可

- 

4．Oracle9i 初始化参数注解

Blank_trimming:

说明：如果值为 TRUE，即使源长度比目标长度（SQL92 兼容）更长，也允许分配数据。

值范围：TRUE | FALSE

默认值：FALSE

serializable:

说明：确定查询是否获取表级的读取锁，以防止在包含该查询的事务处理被提交之前更新任何对象读取。这种操作模式提供可重复的读取，

并确保在同一事务处理种对相同数据的两次查询看到的是相同的值。

值范围：TRUE | FALSE

默认值：FALSE

row_locking:

说明：指定在表已更新或正在更新时是否获取行锁。如果设置为 ALWAYS，只有在表被更新后才获取行锁。如果设置为 INTENT，

只有行锁将用于

SELECT

FOR

UPDATE，但在更新时将获取表锁。

值范围: ALWAYS | DEFAULT | INTENT

默认值: ALWAYS

shared_servers

说明 ： 指定在启动例程后，要为共享服务器环境创建的服务器进程的数量。

值范围: 根据操作系统而定。

默认值 ：1

circuits:

说明 ： 指定可用于入站和出站网络会话的虚拟电路总数。 该参数是构成某个例程的总 SGA 要求的若干参数之一。

默认值 ： 派生: SESSIONS 参数的值 (如果正在使用共享服务器体系结构)；否则为 0。

Mts_multiple_listeners:

说明: 指定多个监听程序的地址是分别指定的，还是用一个 ADDRESS_LIST 字符串指定。如果该值为 TRUE,

MTS_LISTENER_ADDRESS 参数可被指定为:

(ADDRESS_LIST=(ADDRESS=(PROTOCOL=tcp)(PORT=5000)(HOST=zeus))

(ADDRESS=(PROTOCOL=decnet)(OBJECT=outa)(NODE=zeus))

此参数在 8.1.3 版中已废弃。

值范围: TRUE | FALSE

默认值: FALSE

mts_servers:

说明 ： 指定在启动例程后，要为共享服务器环境创建的服务器进程的数量。

值范围: 根据操作系统而定。

默认值 ：1

mts_service:

说明 ： 一个共享服务器参数，用于指定已在调度程序上注册，

用来建立数据库连接的唯一服务名。如果要在没有调度程序的情况下仍能连接到数据库，请将该值设置为与例程名相同。此参数自

8.1.3 版起已废弃。

值范围：根据操作系统而定。

默认值 :0

mts_sessions:

说明 ： 指定允许的共享服务器体系结构用户会话的总数。设置此参数可为专用服务器保留一些用户会话。

值范围：0 到 SESSIONS - 5

默认值 ： 派生：MTS_CIRCUITS 和 SESSIONS - 5 两者中的较小值

shared_server_sessions:

说明 ： 指定允许的共享服务器体系结构用户会话的总数。设置此参数可为专用服务器保留一些用户会话。

值范围：0 到 SESSIONS - 5

默认值 ： 派生：MTS_CIRCUITS 和 SESSIONS - 5 两者中的较小值

mts_max_dispatchers

说明 ： 指定在一个共享服务器环境中可同时运行的调度程序进程的最大数量。

值范围：根据操作系统而定。

默认值 ： 如果已配置了调度程序，则默认值为大于 5 的任何数目或配置的调度程序的数目

mts_max_servers:

说明 ： 指定在一个共享服务器环境中可同时运行的共享服务器进程的最大数量。

值范围：根据操作系统而定。

默认值 ：20

dispatchers:

说明 ： 为设置使用共享服务器的共享环境而设置调度程序的数量和类型。可以为该参数指定几个选项。有关详细信息，

请参阅 “ Oracle8i 管理员指南 ” 和 “ Oracle Net Administrator's

Guide ” 。这是字符串值的一个示例：\''(PROTOCOL=TCP)(DISPATCHERS=3)\''。

值范围：参数的有效指定值。

默认值 ：NULL

max_shared_servers:

说明 ： 指定在一个共享服务器环境中可同时运行的共享服务器进程的最大数量。

值范围：根据操作系统而定。

默认值 ：20

mts_circuits:

说明 ： 指定可用于入站和出站网络会话的虚拟电路总数。 该参数是构成某个例程的总 SGA 要求的若干参数之一。

默认值 ： 派生：SESSIONS 参数的值 (如果正在使用共享服务器体系结构)；否则为 0。

Mts_listener_address:

说明 ： 指定共享服务器的监听程序配置。监听程序进程需要一个监听地址，以便处理系统所用的各个网络协议的连接请求。 除非

MTS_MULTIPLE_LISTENERS=TRUE，否则每个条目都必须有一个独立的相邻值。此参数自 8.1.3

版起已废弃

语法 ：(ADDRESS=(PROTOCOL=tcp)(HOST=myhost)(PORT=7002))

默认值 ：NULL

mts_dispatchers:

说明 ： 为设置使用共享服务器的共享环境而设置调度程序的数量和类型。可以为该参数指定几个选项。有关详细信息，

请参阅 “ Oracle8i 管理员指南 ” 和 “ Oracle Net Administrator's

Guide ” 。这是字符串值的一个示例: \''(PROTOCOL=TCP)(DISPATCHERS=3)\''。

值范围: 参数的有效指定值。

默认值 ：NULL

max_dispatchers:

说明 ： 指定在一个共享服务器环境中可同时运行的调度程序进程的最大数量。

值范围: 根据操作系统而定。

默认值 ： 如果已配置了调度程序，则默认值为大于 5 的任何数目或配置的调度程序的数目

nls_nchar_conv_excp:

说明: (如果值为 TRUE) 当在隐式转换中丢失数据时返回错误的参数。

值范围: FALSE | TRUE

默认值: TRUE

nls_numeric_characters:

说明: 指定将用作组分隔符和小数位的字符。组分隔符就是用来分隔整数位组 (如千，百万等等)

的字符。小数分隔符用来将一个数字的整数部分与小数部分分隔开。其格式是

<decimal_character><group_separator>。

值范围: 任何单字节字符，'+', '-', '<', '>' 除外。

默认值: 从 NLS_TERRITORY 中获得

nls_sort:

说明：指定 ORDER BY 查询的比较顺序。对于二进制排序，ORDER BY 查询的比较顺序是以数值为基础的。对于语言排序，

则需要进行全表扫描，以便将数据按照所定义的语言排序进行整理。

值范围：BINARY 或有效的语言定义名。

默认值：从 NLS_LANGUAGE 中获得

nls_territory：

说明：为以下各项指定命名约定，包括日期和星期的编号，默认日期格式，默认小数点字符和组分隔符，以及默认的 ISO

和本地货币符号。可支持的区域包括美国，法国和日本。有关所有区域的信息，请参阅 Oracle8i National

Language Support Guide。

值范围：任何有效的地区名。

默认值：根据操作系统而定

nls_timestamp_format：

说明：与 NLS_TIME_FORMAT 相似，只不过它设置的是 TIMESTAMP 数据类型的默认值，该数据类型既存储

YEAR，MONTH 和 DAY 这几个日期值，也存储 HOUR，MINUTE 和 SECOND 这几个时间值。

语法：TIMESTAMP '1997-01-31 09:26:50.10' (将值存储为 11 个字节)。

默认值：从 NLS_TERRITORY 中获得

nls_time_format：

说明：指定一个字符串值，设置 TIME 数据类型的默认值，该数据类型包含 HOUR，MINUTE 和 SECOND

这几个日期时间字段。

语法：TIME '09:26:50' (将值存储为 7 个字节)。

默认值：从 NLS_TERRITORY 中获得

nls_time_tz_format：

说明：指定一对值 (UTC,TZD)，设置 TIME WITH TIME ZONE 数据类型的默认值，该数据类型包含

HOUR, MINUTE, SECOND, TIMEZONE_HOUR 和 TIMEZONE_MINUTE

这几个日期时间字段。UTC 是世界时而 TZD 是当地时区。

语法：TIME '09:26:50.20+ 02:00' (将值存储为 9 个字节)。

默认值：从 NLS_TERRITORY 中获得

nls_length_semantics：

说明：使用字节或码点语义来指定新列的创建，如 char, varchar2, clob, nchar, nvarchar2,

nclob 列。各种字符集对字符都有各自的定义。在客户机和服务器上使用同一字符集时,

应以该字符集所定义的字符来衡量字符串。现有的列将不受影响。

值范围：BYTE 或 CHAR。

默认值：nls_length_semantics 的数据库字符集的字符所使用的度量单位。BYTE。

nls_date_format：

说明：指定与 TO_CHAR 和 TO_DATE 函数一同使用的默认日期格式。该参数的默认值由 NLS_TERRITORY

确定。该参数的值可以是包含在双引号内的任何有效的日期格式掩码。例如：''MMM/DD/YYYY''。

值范围：任何有效的日期格式掩码，但不得超过一个固定长度。

默认值：派生

nls_timestamp_tz_format：

说明：与 NLS_TIME_TZ_FORMAT 相似，其中的一对值指定 TIMESTAMP 数据类型的默认值，该类型除存储

YEAR, MONTH 和 DAY 日期值, HOUR, MINUTE 和 SECOND 时间值, 还存储

TIMEZONE_HOUR 和 TIMEZONE_MINUTE。

语法: TIMESTAMP '1997- 01- 31 09:26:50+ 02:00' (将值存储为 13 个字节)。

默认值: 从 NLS_TERRITORY 中获得

nls_language:

说明: 指定数据库的默认语言, 该语言将用于消息, 日期和月份名, AD, BC, AM 和 PM 的符号,

以及默认的排序机制。可支持的语言包括英语, 法语和日语等等。

值范围: 任何有效的语言名。

默认值: 根据操作系统而定

nls_comp:

说明: 在 SQL 语句中, 应避免使用繁琐的 NLS_SORT 进程。正常情况下,

WHERE 子句中进行的比较是二进制的, 但语言比较则需要 NLSSORT 函数。可以使用 NLS_COMP 指定必须根据

NLS_SORT 会话参数进行语言比较。

值范围: Oracle8i National Language Support Guide 中指定的任何有效的 10 字节字符串。

默认值: BINARY

nls_currency:

说明: 为 L 数字格式元素指定用作本地货币符号的字符串。该参数的默认值由 NLS_TERRITORY 确定。

值范围: Oracle8i National Language Support Guide 中指定的任何有效的 10 字节字符串。

默认值: 从 NLS_TERRITORY 中获得

nls_date_language:

说明：指定拼写日期名，月名和日期缩写词（AM, PM, AD, BC）的语言。该参数的默认值是由 NLS_LANGUAGE

指定的语言。

值范围：任何有效的 NLS_LANGUAGE 值。

默认值：NLS_LANGUAGE 的值

nls_dual_currency:

说明：用于覆盖 NLS_TERRITORY 中定义的默认双重货币符号。如果不设置该参数，就会使用默认的双重货币符号；

否则就会启动一个值为双重货币符号的新会话。

值范围：任何有效的格式名。。

默认值：双重货币符号

nls_iso_currency:

说明：为 C 数字格式元素指定用作国际货币符号的字符串。该参数的默认值由 NLS_TERRITORY 确定。

值范围：任何有效的 NLS_TERRITORY 值。

默认值：从 NLS_TERRITORY 中获得

nls_calendar:

说明：指定 Oracle 使用哪种日历系统作为日期格式。例如，如果 NLS_CALENDAR 设置为 'Japanese

Imperial'，那么日期格式为 'E YY-MM-DD'。即：如果日期是 1997 年 5 月 15 日，那么

SYSDATE 显示为 'H 09-05-15'。

值范围：Arabic Hijrah, English Hijrah, Gregorian, Japanese Imperial, Persian, ROC Official (Republic of China) 和 Thai Buddha。

默认值：Gregorian

plsql_native_c_compiler:

说明: 指定用于将生成的 C 文件编译为目标文件的 C 编译程序的完整路径名。此参数是可选的。随每个平台附带的特有的 make

文件中包含此参数的默认值。如果为此参数指定了一个值，则该值将覆盖 make 文件中的默认值。

值范围: C 编译程序的完整路径。

默认值: 无

5．关于 Oracle 数据库的升级(Migration)

1.如果想要升级到 Oracle9i，那么当前的数据库版本必须是 Oracle8.0.6 或者更高版本
2.如果当前数据库是较早的版本，那么必须先升级到 Oracle8 或者 Oracle8i
3.升级方法：一是可以使用 Oracle Data Migration Assistant 工具来完成，而是可以手动升级
4.如果当前数据库中存在 Oracle Parallel Server，那么必须手动升级，因为对于这种情况 Migration Assistant 无法处理
5.详细的升级信息都可以从"Oracle9i Database Migration Release 2 (9.2) Part Number A96530-02"文档中得到

三．Oracle 技巧与提示（来自 Kamus 版主）

1.如何修改数据库的字符集

以下方法只适用于需要修改的字符集是当前字符集的超集，声明：以下方法请谨慎使用，同时一定要在修改字符集以前作当前数据库的完全冷备份。

[转载自 http://www.xray.mpe.mpg.de]
This article describes how one can change the character set of a database.
It should be done with extreme caution having noted the following limitations.

The database character set should only be changed if the characters of the code page of the originating database correspond to the same characters

of the target database, ie. if the database was created with the characterset US7ASCII and it is to be updated to WE8ISO8859P1. Since these have the same encoding scheme for the first 127 bits, changing the character set from US7ASCII to WE8ISO8859P1 will display all characters up to 127 bits as the same character before and after. In addition, in this particular example, if any characters have been entered with the 8th bit set, then updating the database characterset to 8 bit will allow that 8th bit to be displayed. You must not change the characterset from one encoding scheme to another encoding scheme where the code pages do not correspond. This will completely scramble your database. In addition, if case*designer diagrams are stored in the database, this method must not be used. Contact Worldwide Support for further details.

Before attempting to run any of the scripts below, you must take a full cold backup of your database. In addition, the procedure must be thoroughly tested before attempting this on a production instance.

Here's a SQL*PLUS script that allows a database's character set to be changed to a different encoding scheme without having to rebuild the database.

```
set echo off
set verify off

rem The data dictionary table that records the database
rem character set is sys.props$
rem
rem SQL> describe sys.props$
rem Name Null? Type
rem ------------------------------ -------- ----
rem NAME NOT NULL VARCHAR2(30)
rem VALUE$ VARCHAR2(2000)
rem COMMENT$ VARCHAR2(2000)

rem For example:
rem
rem SQL> column c1 format a30
rem SQL> select name c1, value$ c1 from sys.props$;

rem C1 C1
rem ------------------------------ ------------------------------
rem DICT.BASE 2
rem NLS_LANGUAGE AMERICAN
rem NLS_TERRITORY AMERICA
rem NLS_CURRENCY $
```

```
rem NLS_ISO_CURRENCY AMERICA
rem NLS_NUMERIC_CHARACTERS .,
rem NLS_DATE_FORMAT DD-MON-YY
rem NLS_DATE_LANGUAGE AMERICAN
rem NLS_CHARACTERSET WE8DEC
rem NLS_SORT BINARY
rem GLOBAL_DB_NAME NLSV7.WORLD

rem NLS_CHARACTERSET can be changed by updating its value, for example:

rem update sys.props$
rem set value$ = 'WE8ISO8859P1'
rem Where name = 'NLS_CHARACTERSET';

rem The database has to be shutdown and restarted before the change
rem becomes effective.

rem It is very important to specify the character set name correctly.
rem IMPORTANT NOTE
rem ==============
rem If NLS_CHARACTERSET is updated to an invalid value, it will not then
rem be possible to restart the database once it has been shutdown.
rem To recover, it will be necessary to re-create the database, since it
rem cannot be restarted to correct the invalid NLS_CHARACTERSET entry.

rem The character set name should be in uppercase.
rem The new value is not effective until the database has been shutdown and
rem restarted.
rem
rem A suggested procedure is as follows, and can be done by running this
rem script from SQL*Plus when logged into the SYSTEM account.
rem
rem USAGE : SQL> start ch_db.sql <character set>
rem
rem where <character set> is the desired database character set
rem

Prompt First check that the character set name is valid.

set echo on

select convert('a','&1','us7ascii') from dual;

set echo off
```

```
prompt If this select statement returns error ORA-01482, then the
prompt specified character set name is not valid for this installation.
prompt Abort the procedure now with Control-c

prompt To continue, press return
accept ans CHAR

Prompt Check the current value of database character set.

column c1 format a30
select name c1, value$ c1 from sys.props$
where name = 'NLS_CHARACTERSET';

prompt To continue, press return

Prompt Update to new character set

update sys.props$
set value$ = upper('&1')
where name = 'NLS_CHARACTERSET';

set echo off

prompt To continue, press return
accept ans CHAR

Prompt Check the new value of database character set

select name c1, value$ c1 from sys.props$
where name = 'NLS_CHARACTERSET';

Prompt If the value is updated as required, press return to continue and
Prompt then manually type COMMIT; to commit the change. Then shutdown and
Prompt restart the database.
Prompt
Prompt If the value is not updated as required, press return to continue and
Prompt than manually type ROLLBACK; to prevent the change.

prompt To continue, press return
accept ans CHAR
----------------------------------------->
```

2.如何查看 Control File 中保存的内容

Control File 是二进制文件，用普通的方法很难知道其中到底保存了什么内容，但是 Oracle 却提供了一个 SQL*PLUS 命令来将 Control File 的内容 dump 到文本文件中。
方法如下：
以 SYSDBA 身份登入 SQL*PLUS
SQL> oradebug setmypid
SQL> oradebug dump controlf 3

将把 control file dump 到 USER_DUMP_DEST 初始化参数指定的目录下。
其中 3 为 dump level。
level 的解释如下：
1 ：only the file header
2 ：just the file header, the database info record, and checkpoint progress records
3 ：all record types, but just the earliest and latest records for circular reuse record types
4 ：as above, but includes the 4 most recent records for circular reuse record types
5+ ：as above, but the number of circular reuse records included doubles with each level

3.Oracle9i(Version 9.2)SYS_CONTEXT 函数的用法以及同 USERENV 函数的比较

Oracle9i(Version 9.2)SYS_CONTEXT 函数的用法

这个函数在写一些触发器，函数的时候非常有用处。

用法：SELECT sys_context('USERENV', '<parameter>') FROM dual;

第二个参数的可选值：
AUTHENTICATION_DATA
Data being used to authenticate the login user. For X.503 certificate authenticated sessions, this field returns the context of the certificate in HEX2 format.
Note: You can change the return value of the AUTHENTICATION_DATA attribute using the length parameter of the syntax. Values of up to 4000 are accepted. This is the only attribute of USERENV for which Oracle implements such a change.

AUTHENTICATION_TYPE
How the user was authenticated:
DATABASE: username/password authentication
OS: operating system external user authentication
NETWORK: network protocol or ANO authentication
PROXY: OCI proxy connection authentication

BG_JOB_ID
Job ID of the current session if it was established by an Oracle background process. Null if the session was not established by a background process.

CLIENT_INFO
Returns up to 64 bytes of user session information that can be stored by an application using the DBMS_APPLICATION_INFO package.

CURRENT_SCHEMA
Name of the default schema being used in the current schema. This value can be changed during the session with an ALTER SESSION SET CURRENT_SCHEMA statement.

CURRENT_SCHEMAID
Identifier of the default schema being used in the current session.

CURRENT_USER
The name of the user whose privilege the current session is under.

CURRENT_USERID
User ID of the user whose privilege the current session is under.

DB_DOMAIN
Domain of the database as specified in the DB_DOMAIN initialization parameter.

DB_NAME
Name of the database as specified in the DB_NAME initialization parameter.

ENTRYID
The available auditing entry identifier. You cannot use this option in distributed SQL statements. To use this keyword in USERENV, the initialization parameter AUDIT_TRAIL must be set to true.

EXTERNAL_NAME
External name of the database user. For SSL authenticated sessions using v.503 certificates, this field returns the distinguished name (DN) stored in the user certificate.

FG_JOB_ID
Job ID of the current session if it was established by a client foreground process. Null if the session was not established by a foreground process.

HOST
Name of the host machine from which the client has connected.

INSTANCE
The instance identification number of the current instance.

IP_ADDRESS
IP address of the machine from which the client is connected.

ISDBA
TRUE if you are logged on as SYS.

LANG
The ISO abbreviation for the language name, a shorter form than the existing 'LANGUAGE' parameter.

LANGUAGE
The language and territory currently used by your session, along with the database character set, in the form:language_territory.characterset.

NETWORK_PROTOCOL
Network protocol being used for communication, as specified in the 'PROTOCOL=protocol' portion of the connect string.

NLS_CALENDAR
The current calendar of the current session.

NLS_CURRENCY
The currency of the current session.

NLS_DATE_FORMAT
The date format for the session.

NLS_DATE_LANGUAGE
The language used for expressing dates.

NLS_SORT BINARY
or the linguistic sort basis.

NLS_TERRITORY
The territory of the current session.

OS_USER
Operating system username of the client process that initiated the database session.

PROXY_USER

Name of the database user who opened the current session on behalf of SESSION_USER.

PROXY_USERID
Identifier of the database user who opened the current session on behalf of SESSION_USER.

SESSION_USER
Database user name by which the current user is authenticated. This value remains the same throughout the duration of the session.

SESSION_USERID
Identifier of the database user name by which the current user is authenticated.

SESSIONID
The auditing session identifier. You cannot use this option in distributed SQL statements.

TERMINAL
The operating system identifier for the client of the current session. In distributed SQL statements, this option returns the identifier for your local session. In a distributed environment, this is supported only for remote SELECT statements, not for remote INSERT, UPDATE, or DELETE operations.
(The return length of this parameter may vary by operating system.)

对于这个函数有个很奇怪的结果，下面是对认证类型的选择。
C:\>sqlplus "sys/dba@ora921.hl3 as sysdba"

SQL*Plus: Release 9.2.0.4.0 - Production on 星期四 10月 23 21:08:21 2003

Copyright (c) 1982, 2002, Oracle Corporation. All rights reserved.

连接到:
Oracle9i Enterprise Edition Release 9.2.0.4.0 - Production
With the Partitioning, OLAP and Oracle Data Mining options
JServer Release 9.2.0.4.0 - Production

SQL> SELECT sys_context('USERENV','AUTHENTICATION_TYPE') FROM dual;

SYS_CONTEXT('USERENV','AUTHENTICATION_TYPE')

OS

SQL>

可以明显地看出这个认证明明是应该通过 DATABASE 的，但是却显示 OS。
开始我以为是因为我的用户在那个服务器上也有相同的用户（NT），所以 oracle 选择了操作系统认证，后来换了用户登陆，并且登陆到 linux 上的 oracle，发现结果仍然是 OS。
难道是 bug？

摘自 DBASUPPORT.COM
全文连接如下，有第二部分没有摘抄：
http://forums.dbasupport.com/oracle...s_context.shtml

USERENV vs. SYS_CONTEXT
First, let's consider what information the deprecated USERENV function can provide. In this example, I need to determine whether I am logged on as the DBA; what the instance number of the current database is; what terminal I am currently logged on from; and what the NLS Territory parameters have been set to. Here is an example utilizing USERENV to return this information:

```
SQL> TTITLE CENTER "Example of USERENV() Function"
SQL> COLUMN amidba FORMAT A9 HEADING "Am I DBA?"
SQL> COLUMN instance FORMAT 99999 HEADING "Inst|ID"
SQL> COLUMN terminal FORMAT A16 HEADING "Terminal"
SQL> COLUMN language FORMAT A32 HEADING "NLS Language Parameters"
SQL> SELECT
2 USERENV('ISDBA') amidba
3 ,USERENV('INSTANCE') instance
4 ,USERENV('TERMINAL') terminal
5 ,USERENV('LANGUAGE') language
6 FROM DUAL;
Example of USERENV() Function
Inst
Am I DBA? ID Terminal NLS Language Parameters
--------- ------ --------------- ------------------------------
FALSE 1 MAIN_CONSOLE AMERICAN_AMERICA.WE8MSWIN1252
```

And here's its counterpart using SYS_CONTEXT against the USERENV namespace:

```
SQL> TTITLE CENTER "Example of SYS_CONTEXT() Function"
SQL> COLUMN amidba FORMAT A9 HEADING "Am I DBA?"
SQL> COLUMN instance FORMAT A6 HEADING "Inst|ID"
SQL> COLUMN terminal FORMAT A16 HEADING "Terminal"
SQL> COLUMN language FORMAT A32 HEADING "NLS Language Parameters"
```

```
SQL>
SQL> SELECT
2 SYS_CONTEXT('USERENV', 'ISDBA') amidba
3 ,SYS_CONTEXT('USERENV', 'INSTANCE') instance
4 ,SYS_CONTEXT('USERENV', 'TERMINAL') terminal
5 ,SYS_CONTEXT('USERENV', 'LANGUAGE') language
6 FROM DUAL;

Example of SYS_CONTEXT() Function
Inst
Am I DBA? ID Terminal NLS Language Parameters
--------- ------ --------------- ------------------------------
FALSE 1 MAIN_CONSOLE AMERICAN_AMERICA.WE8MSWIN1252
```

Expanded Session Information

SYS_CONTEXT is more than just a replacement for USERENV, as the next example shows. There are over two dozen namespace attributes that can be queried. This next example shows how to return:

the session's IP address
the session's host machine name
the network protocol in use
the current user and schema
whether the session is a foreground task

Here is a sample of some session and user-specific information that can be obtained in one function call:

```
SQL> TTITLE CENTER "More Session-Level Information from SYS_CONTEXT()"
SQL> COLUMN ipaddr FORMAT A15 HEADING "IP Address"
SQL> COLUMN host FORMAT A20 HEADING "Host"
SQL> COLUMN netprtc FORMAT A8 HEADING "Network|Protocol"
SQL> COLUMN curruser FORMAT A8 HEADING "Current|User"
SQL> COLUMN currschema FORMAT A8 HEADING "Current|Schema"
SQL> COLUMN fgjob FORMAT A4 HEADING "FG|Job?"
SQL> COLUMN bgjob FORMAT A4 HEADING "BG|Job?"
SQL>
SQL> SELECT
2 SYS_CONTEXT('USERENV', 'IP_ADDRESS', 15) ipaddr
3 ,SYS_CONTEXT('USERENV', 'HOST', 16) host
4 ,SYS_CONTEXT('USERENV', 'NETWORK_PROTOCOL', 8) netprtc
5 ,SYS_CONTEXT('USERENV', 'CURRENT_USER', 8) curruser
6 ,SYS_CONTEXT('USERENV', 'CURRENT_SCHEMA', 8) currschema
7 ,SYS_CONTEXT('USERENV', 'FG_JOB_ID', 4) fgjob
```

```
8 ,SYS_CONTEXT('USERENV', 'BG_JOB_ID', 4) bgjob
9 FROM DUAL;

More Session-Level Information from SYS_CONTEXT()
Network Current Current FG BG
IP Address Host Protocol User Schema Job? Job?
--------------- -------------------- -------- -------- -------- ---- ----
198.63.66.124 WORKGROUP\MYCONSOLE tcp HR HR 0
```

With the exception of one parameter for the USERENV namespace - AUTHENTICATION_DATA, which will return the data being used to authenticate the login user - the maximum length of the VARCHAR2 string that SYS_CONTEXT returns is 255 characters, depending upon the parameter chosen. However, note from the previous examples that the SYS_CONTEXT function can also take one other argument after the specified parameter to limit the maximum length of the returned value.

oracle 官方文档的链接：
USERENV
http://www.lc.leidenuniv.nl/awcours...nctions162a.htm

SYS_CONTEXT
http://www.lc.leidenuniv.nl/awcours...22a.htm#1038178

附件二：ORACLE 体系结构

(Architecture of ORACLE)

第一部分：ORACLE8i 体系结构

第一章. 概要

在本章里你可以了解以下内容

1、 理解 ORACLE 实例的组成
2、 理解 ORACLE 数据库的组成
3、 理解 ORACLE 内存结构的组成

4、 理解后台进程的作用与分工
5、 理解数据库的物理文件与对应的逻辑结构
6、 理解 ORACLE 的整体构架

第二章. 理解 ORACLE 实例

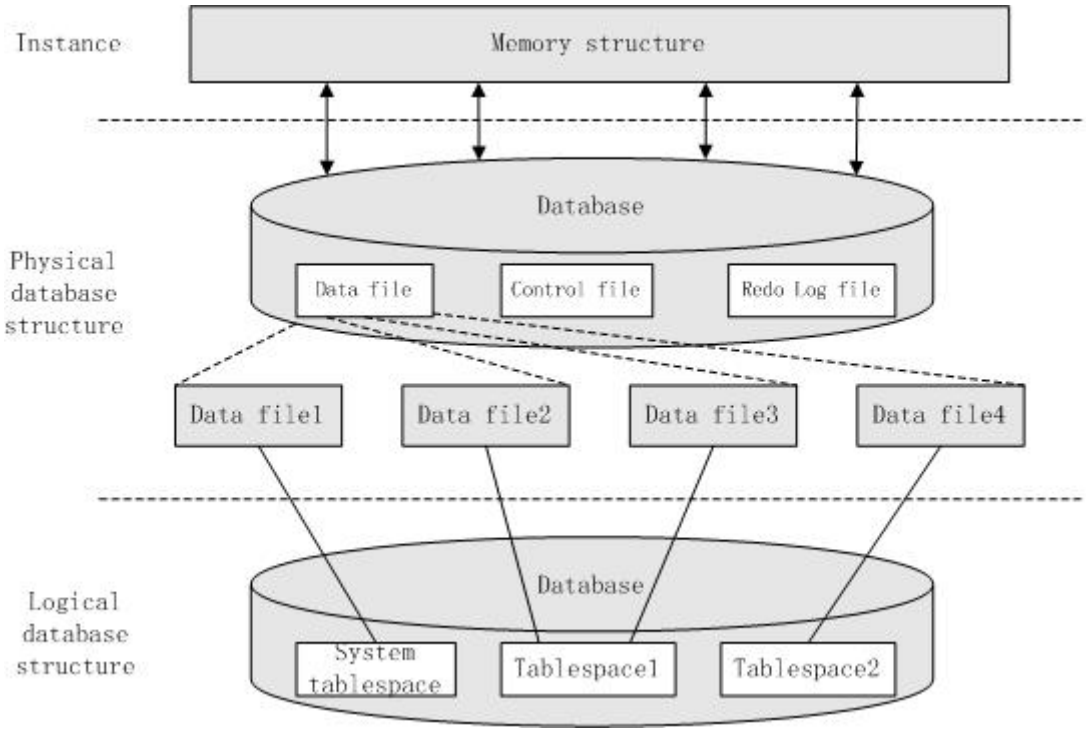
2.1 ORACLE SERVER

ORACLE 是一个可移植的数据库——它在相关的每一个平台上都可以使用，即所谓的跨平台特性。在不同的操作系统上也略有差别，如在 UNIX/LINUX 上，ORACLE 是多个进程实现的，每一个主要函数都是一个进程；而在 Windows 上，则是一个单一进程，但是在该进程中包含多个线程。但是从整体构架上来看，ORACLE 在不同的平台上是一样的，如内存结构、后台进程、数据的存储。

一个运行着的 ORACLE 数据库就可以看成是一个 ORACLE SERVER，该 SERVER 由数据库(Database)和实例(Instance)组成，在一般的情况下一个 ORACLE SERVER 包含一个实例和一个与之对应的数据库，但是在特殊情况下，如 8i 的 OPS，9i 的 RAC，一个 SERVER 中一个数据库可以对应多个实例。

一系列物理文件（数据文件，控制文件，联机日志等）的集合或与之对应的逻辑结构（表空间，段等）被称为数据库，简单的说，就是一系列与磁盘有关系的物理文件的组成。**ORACLE 内存结构和后台进程被成为数据库的实例**，一个实例最多只能安装(Mount)和打开(Open)在一个数据库上，负责数据库的相应操作并与用户交互。

实例与数据库的关系如下图所示：

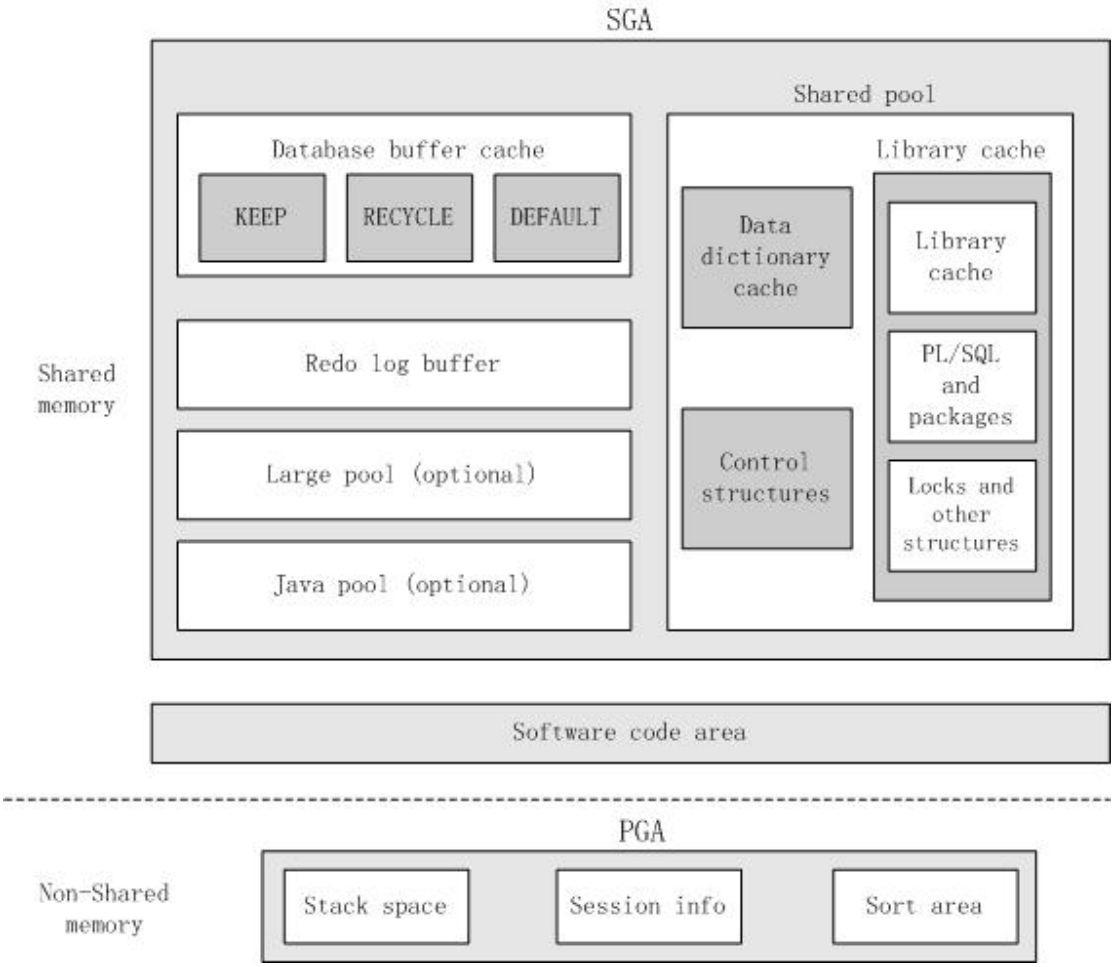


图一 ORACLE SERVER

2.2 ORACLE 内存结构 (Memory structure)

2.2.1 内存结构的组成

Oracle 内存结构主要可以分共享内存区与非共享内存区，共享内存区主要包含 SGA(System Global Area)，非共享内存区主要由 PGA(Program Global Area)组成，可以用如下图形表示。



图二 ORACLE MEMOERY STRUCTRUE

2.2.2 全局共享区 System Global Area(SGA)

System Global Area 是一块巨大的共享内存区域，他被看做是 Oracle 数据库的一个大缓冲池，这里的数据可以被 ORACLE 的各个进程共用。其大小可以通过如下语句查看：

```
SQL> select * from v$sga;
NAME VALUE
---------- ---------
Fixed Size 39816
Variable Size 259812784
```

Database Buffers 1.049E+09
Redo Buffers 327680
更详细的信息可以参考 V$sgastat、V$buffer_pool
主要包括以下几个部分：

2.2.2.1 共享池(Shared pool)

共享池是 SGA 中最关键的内存片段，特别是在性能和可伸缩性上。一个太小的共享池会扼杀性能，使系统停止，太大的共享池也会有同样的效果，将会消耗大量的 CPU 来管理这个共享池。不正确的使用共享池只会带来灾难。共享池主要又可以分为以下两个部分：

1、SQL 语句缓冲(Library Cache)

当一个用户提交一个 SQL 语句，Oracle 会将这句 SQL 进行分析(parse)，这个过程类似于编译，会耗费相对较多的时间。在分析完这个 SQL，Oracle 会把他的分析结果给保存在 Shared pool 的 Library Cache 中，当数据库第二次执行该 SQL 时，Oracle 自动跳过这个分析过程，从而减少了系统运行的时间。这也是为什么第一次运行的 SQL 比第二次运行的 SQL 要慢一点的原因。

下面举例说明 parse 的时间

```
SQL>Startup
SQL> select count(*) from usertable;
COUNT(*)
----------
243
Elapsed: 00:00:00.08
```

这是在 Share_pool 和 Data buffer 都没有数据缓冲区的情况下所用的时间

```
SQL> alter system flush SHARED_POOL;
System altered.
```

清空 Share_pool，保留 Data buffer

```
SQL> select count(*) from usertable;
COUNT(*)
-----------------
243
Elapsed: 00:00:00.02
SQL> select count(*) from usertable;
COUNT(*)
----------------
243
Elapsed: 00:00:00.00
```

从两句 SQL 的时间差上可以看出该 SQL 的 Parse 时间约为 00:00:00.02
对于保存在共享池中的 SQL 语句，可以从 V$Sqltext、v$Sqlarea 中查询到，对于编程者来说，要尽量提高语句的重用率，减少语句的分析时间。一个设计的差的应用程序可以毁掉整个数据库的 Share pool，提高 SQL 语句的重用率必须先养成良好的变成习惯，尽量使用 Bind 变量。

2、数据字典缓冲区(Data Dictionary Cache)

显而易见，数据字典缓冲区是 ORACLE 特地为数据字典准备的一块缓冲池，供 ORACLE

内部使用，没有什么可以说的。

2.2.2.2 块缓冲区高速缓存(Database Buffer Cache)

这些缓冲是对应所有数据文件中的一些被使用到的数据块。让他们能够在内存中进行操作。在这个级别里没有系统文件,，户数据文件，临时数据文件，回滚段文件之分。也就是任何文件的数据块都有可能被缓冲。数据库的任何修改都在该缓冲里完成，并由 DBWR 进程将修改后的数据写入磁盘。

这个缓冲区的块基本上在两个不同的列表中管理。一个是块的“脏”表(Dirty List)，需要用数据库块的书写器(DBWR)来写入，另外一个是不脏的块的列表(LRU List)，一般的情况下，是使用最近最少使用(Least Recently Used,LRU)算法来管理。

块缓冲区高速缓存又可以细分为以下三个部分（Default pool,Keep pool,Recycle pool）。如果不是人为设置初始化参数(Init.ora)，ORACLE 将默认为 Default pool。

由于操作系统寻址能力的限制，不通过特殊设置，在 32 位的系统上，块缓冲区高速缓存最大可以达到 1.7G，在 64 位系统上，块缓冲区高速缓存最大可以达到 10G。

2.2.2.3 重做日志缓冲区(Redo log buffer)

重做日志文件的缓冲区，对数据库的任何修改都按顺序被记录在该缓冲，然后由 LGWR 进程将它写入磁盘。这些修改信息可能是 DML 语句，如(Insert,Update,Delete)，或 DDL 语句，如(Create,Alter,Drop 等)。

重做日志缓冲区的存在是因为内存到内存的操作比较内存到硬盘的速度快很多，所以重作日志缓冲区可以加快数据库的操作速度，但是考虑的数据库的一致性与可恢复性，数据在重做日志缓冲区中的滞留时间不会很长。所以重作日志缓冲区一般都很小，大于 3M 之后的重作日志缓冲区已经没有太大的实际意义。

2.2.2.4 Java 程序缓冲区(Java Pool)

Java 的程序区，Oracle 8I 以后，Oracle 在内核中加入了对 Java 的支持。该程序缓冲区就是为 Java 程序保留的。如果不用 Java 程序没有必要改变该缓冲区的默认大小。

2.2.2.5 大池(Large Pool)

大池的得名不是因为大，而是因为它用来分配大块的内存，处理比共享池更大的内存，在 8.0 开始引入。

下面对象使用大池：

1、 MTS——在 SGA 的 Large Pool 中分配 UGA
2、 语句的并行查询(Parallel Executeion of Statements)——允许进程间消息缓冲区的分配，用来协调并行查询服务器
3、 备份(Backup)——用于 RMAN 磁盘 I/O 缓存

2.2.3 程序共享区 Program Global Area(PGA)

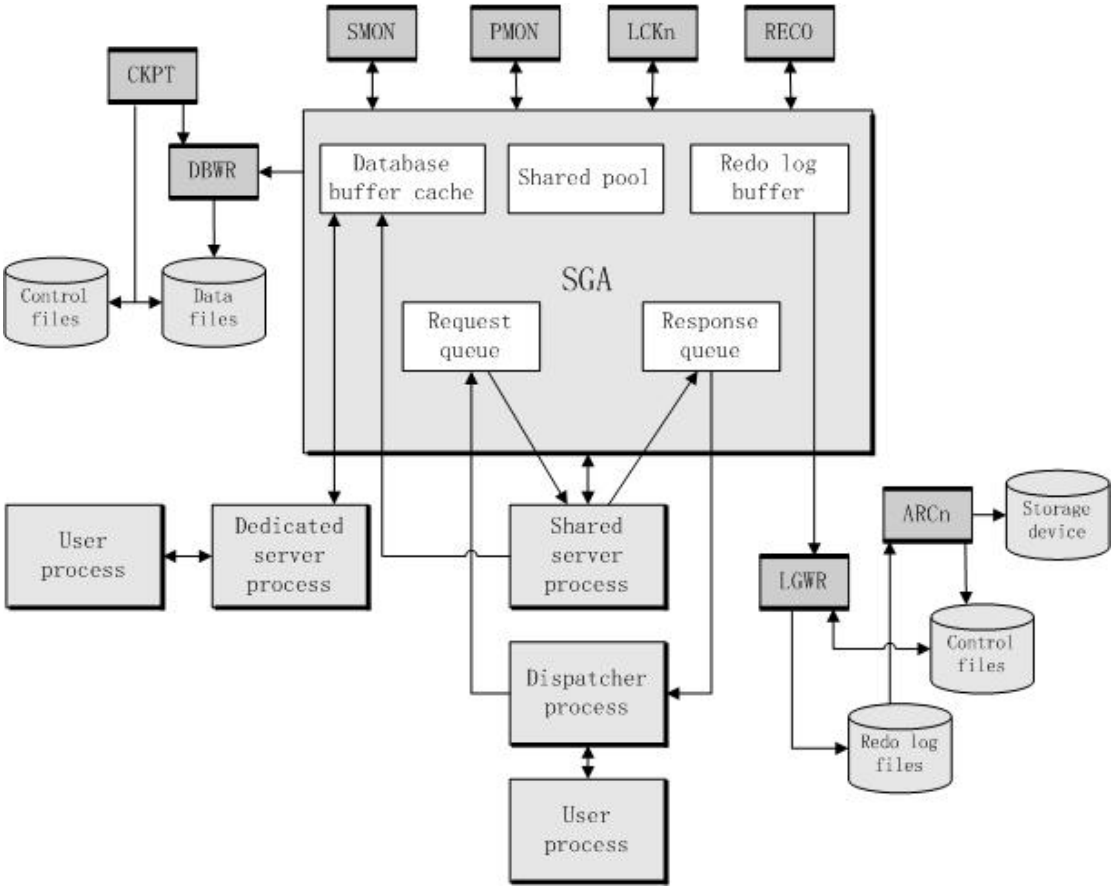
Program Global Area(PGA)是用来保存与用户进程相关的内存段，PGA 总是由进程或线程在本地分配，其它进程与线程无法访问。

User Global Area(UGA)实际上是会话的状态，它是会话必须始终能够得到的内存。对于专用服务器进程，UGA 在 PGA 中分配。对于多线程进程，UGA 在 Large pool 中分配。

PGA/UGA 一般保存了用户的变量、权限、堆栈、排序（Sort）空间等信息。影响 PGA/UGA 最大的也就是 Sort 信息，由初始化参数 sort_area_size 决定，由于 Sort 信息分配在 UGA 中，所以在共享服务器中能更好的利用内存。

2.3 后台进程 (Background process)

后台进程是 Oracle 的程序，用来管理数据库的读写，恢复和监视等工作。Server Process 主要是通过他和 user process 进行联系和沟通，并由他和 user process 进行数据的交换。在 Unix 机器上，Oracle 后台进程相对于操作系统进程，也就是说，一个 Oracle 后台进程将启动一个操作系统进程；在 Windows 机器上，Oracle 后台进程相对于操作系统线程，打开任务管理器，我们只能看到一个 ORACLE.EXE 的进程，但是通过另外的工具，就可以看到包含在这里进程中的线程。后台进程与其它结构的关系如图所示：



图三 ORACLE BACKGROUP PROCESS

在 Unix 上可以通过如下方法查看后台进程：

ps –ef | grep ora_

ps -ef | grep ora_ | grep XCLUAT
oracle 29431 1 0 Sep 02 ? 2:02 ora_dbwr_SID
oracle 29444 1 0 Sep 02 ? 0:03 ora_ckpt_SID
oracle 29448 1 0 Sep 02 ? 2:42 ora_smon_SID
oracle 29442 1 0 Sep 02 ? 3:25 ora_lgwr_SID
oracle 29427 1 0 Sep 02 ? 0:01 ora_pmon_SID
Oracle 系统有 5 个基本进程他们是
DBWR(数据文件写入进程)
LGWR(日志文件写入进程)
SMON(系统监护进程)
PMON(用户进程监护进程)
CKPT(检查点进程,同步数据文件，日志文件,控制文件)

2.3.1 数据写进程 DBWR

将修改过的数据缓冲区的数据写入对应数据文件
维护系统内的空缓冲区
这里指出几个容易错误的概念：
· 当一个更新提交后，DBWR 把数据写到磁盘并返回给用户提交完成。
· DBWR 会触发 CKPT 后台进程
· DBWR 不会触发 LGWR 进程
上面的概念都是错误的。
DBWR 是一个很底层的工作进程，他批量的把缓冲区的数据写入磁盘。和任何前台用户的进程几乎没有什么关系，也不受他们的控制。至于 DBWR 会不会触发 LGWR 和 CKPT 进程，我们将在下面几节里讨论。
DBWR 工作的主要条件如下
· DBWR 超时
· 系统中没有多的空缓冲区用来存放数据
· CKPT 进程触发 DBWR 等

2.3.2 日志写进程 LGWR

将重做日志缓冲区的数据写入重做日志文件，LGWR 是一个必须和前台用户进程通信的进程。当数据被修改的时候，系统会产生一个重做日志并记录在重做日志缓冲区内。这个重做日志可以类似的认为是以下的一个结构:
SCN=000000001000
数据块 ID
对象 ID=0801
数据行=02
修改后的数据=0011
提交的时候，LGWR 必须将被修改的数据的重做日志缓冲区内数据写入日志数据文件，然后再通知前台进程提交成功，并由前台进程通知用户。从这点可以看出 LGWR 承担了维护系统数据完整性的任务。

LGWR 工作的主要条件如下

· 用户提交

· 有 1/3 重做日志缓冲区未被写入磁盘

· 有大于 1M 重做日志缓冲区未被写入磁盘

· 超时

· DBWR 需要写入的数据的 SCN 号大于 LGWR 记录的 SCN 号，DBWR 触发 LGWR 写入

2.3.3 系统监控 SMON

工作主要包含

· 清除临时空间

· 在系统启动时，完成系统实例恢复

· 聚结空闲空间

· 从不可用的文件中恢复事务的活动

· OPS 中失败节点的实例恢复

· 清除 OBJ$表

· 缩减回滚段

· 使回滚段脱机

2.3.4 进程监控 PMON

主要用于清除失效的用户进程，释放用户进程所用的资源。如 PMON 将回滚未提交的工作，释放锁，释放分配给失败进程的 SGA 资源。

2.3.5 检查点进程 CKPT

同步数据文件，日志文件和控制文件，由于 DBWR/LGWR 的工作原理，造成了数据文件，日志文件，控制文件的不一至，这就需要 CKPT 进程来同步。CKPT 会更新数据文件/控制文件的头信息。

CKPT 工作的主要条件如下

· 在日志切换的时候

· 数据库用 immediate ,transaction , normal 选项 shutdown 数据库的时候

· 根据初始话文件 LOG_CHECKPOINT_INTERVAL、LOG_CHECKPOINT_TIMEOUT、FAST_START_IO_TARGET 的设置的数值来确定

· 用户触发

以下进程的启动需要手工配置

2.3.6 归档进程 ARCH

当数据库以归档方式运行的时候，Oracle 会启动 ARCH 进程，当重做日志文件被写满时，日志文件进行切换，旧的重做日志文件就被 ARCH 进程复制到一个/多个特定的目录/

远程机器。这些被复制的重做日志文件被叫做归档日志文件。

2.3.7 分布式恢复 RECO

负责解决分布事物中的故障。Oracle 可以连接远程的多个数据库，当由于网络问题，有些事物处于悬而未决的状态。RECO 进程试图建立与远程服务器的通信，当故障消除后，RECO 进程自动解决所有悬而未决的会话。

2.3.8 服务进程 Server Process

服务进程的分类
· 专用服务进程(Dedicated Server Process)
一个服务进程对应一个用户进程
· 共享服务进程(MultiTreaded Server Process)
一个服务进程对应多个用户进程，轮流为用户进程服务。

2.3.9 用户进程 User Process

在客户端，负责将用户的 SQL 语句传递给服务进程，并从服务器段拿回查询数据。

2.4 一个贯穿数据库全局的概念---- 系统改变号 SCN(System Change Number)

系统改变号，一个由系统内部维护的序列号。当系统需要更新的时候自动增加，他是系统中维持数据的一致性和顺序恢复的重要标志。
运行以下语句可以得到系统 SCN 号

```
SQL> select max(ktuxescnw * power(2, 32) + ktuxescnb) scn from x$ktuxe;
      SCN
----------
   31014
```

SCN 有如下特点:

a. 查询语句不会使 SCN 增加，就算是同时发生的更新，数据库内部对应的 SCN 也是不同的。这样一来就保证了数据恢复时候的顺序。

b. 维持数据的一致性，当一个查询执行的时候，他会先从系统中得到一个当前的 SCN 号，在他查找数据的同时，他会检查每个数据行和他对应的 SCN 号，只有那些不比他的 SCN 号大的行才能从对应用户数据文件的缓冲区内取出，而那些大于他 SCN 号的行，就应该从回滚段数据文件的缓冲中取出。

实例分析:
一个查询返回以下 5 行

```
ID Name
-------------------------------------
```

1 ShangHai
2 Beijing
3 Gugangzhou
4 ShenZhen
5 HanZhou

用户 A 从 12:00 开始运行，到 12:05 结束在 12:01 用户 B 执行了一条 Update 语句，更新了 ID 是 2 的那条记录把 Beijing 该成了 Tianjing.并提交，这时候用户 A 的那个查询是不会出现 Tianjing 的记录。12:00 查询时候的 SCN 是 N 然后用户 B 的更新使得系统的 SCN 变成 N+1 当用户 A 查询到 ID=2 的记录的时候发现他的 SCN 已经大于查询开始时候的 SCN，他就会在回滚段数据缓冲中找到 SCN=N 的那条记录，并把它返回。

第三章. ORACLE 数据库（Database）

3.1 物理结构——物理操作系统文件的集合.

3.1.1 控制文件 （Control files）

参数文件 init.ora 记录了控制文件的位置，控制文件是一个非常小的二进制文件，最大可以增长到 64MB，控制文件包括如下主要信息

· 数据库的名字，检查点信息，数据库创建的时间戳
· 所有的数据文件，联机日志文件，归档日志文件信息
· 备份信息等

有了这些信息，Oracle 就知道那些文件是数据文件，现在的重做日志文件是哪些，这些都是系统启动和运行的基本条件，所以他是 Oracle 运行的根本。如果没有控制文件系统是不可能启动的。控制文件是非常重要的，一般采用多个镜相复制来保护控制文件，或采用 RAID 来保护控制文件。控制文件的丢失，将使数据库的恢复变的很复杂。

控制文件信息可以从 V$Controlfile 中查询获得

3.1.2 数据文件（Data files）

数据文件的详细信息记载在控制文件中
可以通过如下方式查看数据文件
SQL> select name from v$datafile;
NAME

/u05/dbf/PROD/system_01.dbf
/u06/dbf/PROD/temp_01.dbf
/u04/dbf/PROD/users_01.dbf
/u09/dbf/PROD/rbs_01.dbf
/u06/dbf/PROD/applsys_indx_01.dbf

/u05/dbf/PROD/applsys_data_01.dbf

数据文件是 ORACLE 中最重要的物理文件，直接记录了用户数据。按照使用上的不同，可以把数据文件分成如下几类：

· 系统数据文件
· 回滚数据文件
· 临时数据文件
· 用户数据文件

以上各类文件分别属于不同性质的表空间，在以下的逻辑结构中，将进一步说明该类型文件的作用。

3.1.3 重做日志文件（Redo files）

用户对数据库进行的任何操作都会记录在重做日志文件。在了解重做日志之前必须了解重做日志的两个概念，重做日志组和重做日志组成员(Member)，一个数据库中至少要有两个日志组文件，一组写完后再写另一组，即轮流写。每个日志组中至少有一个日志成员，一个日志组中的多个日志成员是镜相关系，有利于日志文件的保护，因为日志文件的损坏，特别是当前联机日志的损坏，对数据库的影响是巨大的。

联机日志组的交换过程叫做切换，需要特别注意的是，日志切换在一个优化效果不好的数据库中会引起临时的“ 挂起 ”。挂起大致有两种情况：

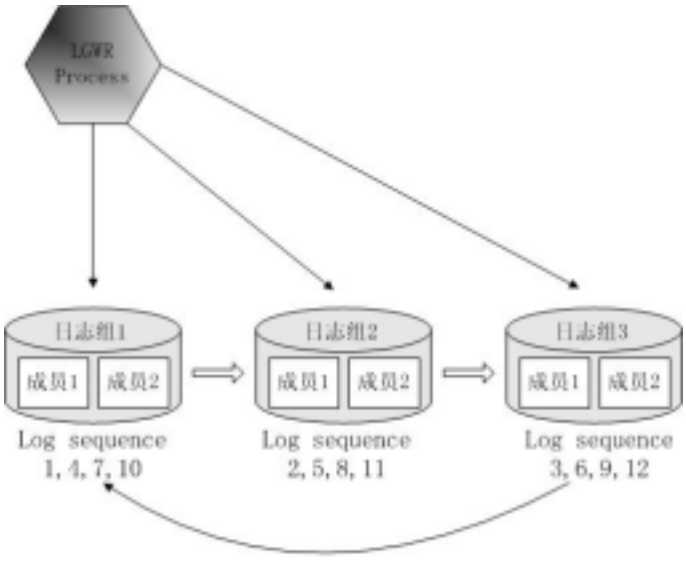
· 在归档情况下，需要归档的日志来不及归档，而联机日志又需要被重新利用
· 检查点事件还没有完成（日志切换引起检查点），而联机日志需要被重新利用

解决这种问题的常用手段是：

i.增加日志组

ii.增大日志文件成员大小

一个包含三个日志组，每个日志组两个成员的联机日志组成与运行大致如图所示：



图四 REDO LOG

通过 v$log 可以查看日志组，v$logfile 可以查看具体的成员文件。

3.1.4 归档日志文件（Archived files）

Oracle 可以运行在两种模式之中，归档模式和不归档模式。如果不用归档模式，当然，你就不会有归档日志，但是，你的系统将不会是一个实用系统，特别是不能用于生产系统，因为你可能会丢失数据。但是在归档模式中，为了保存用户的所有修改，在重做日志文件切换后和被覆盖之间系统将他们另外保存成一组连续的文件系列，该文件系列就是归档日志文件。

有人或许会说，归档日志文件占领我大量的硬盘空间，其实，具体想一想，你是愿意浪费一点磁盘空间来保护你的数据，还是愿意丢失你的数据呢？显而义见，我们需要保证我们的数据的安全性。其实，归档并不是一直占领你的磁盘空间，你可以把她备份到磁带上，或则删除上一次完整备份前的所有日志文件。

通过 v$archived_log 和 v$log_history 可以查看归档日志文件的信息。

3.1.5 初始化参数文件（Parameter file）

initSID.ora 或 init.ora 文件，因为版本的不一样，其位置也可能会不一样。在 8i 中，通常位于$ORACLE_HOME/admin/<SID>/Pfile 下。在 9i 以下参数文件是一个纯文本文件，可以用文本编辑器打开修改，从 9i 开始，多了一个叫 spfile 的参数文件，以二进制方式保存。

初始化文件记载了许多数据库的启动参数，如内存，控制文件，进程数等，在数据库启动的时候加载（Nomount 时加载），初始化文件记录了很多重要参数，对数据库的性能影响很大，如果没有它，数据库将无法启动。在 9i 以前，对参数文件的修改，必须重新启动数据库才能使参数生效，从 9i 开始，可以用命令来修改 spfile 文件的内容了。

参数文件中的参数不是一直一成不变的，随着版本的不同而不同。大多数参数，如 Db_block_size 的寿命就很长，其它很多参数随着版本的改变就被废弃了。除了文档记录的参数外，ORACLE 还支持很多内部参数，当然，这些参数是不建议被使用的。

通过 v$parameter 视图可以查询当前的参数设置。

3.1.6 其他文件

i．密码文件

用于 Oracle 的具有 sysdba 权限用户的认证，在 9i 以前主要指 Internal 用户，从 9i 开始已经取消了这个用户。密码文件的密码可以通过 ORAPWD 命令来修改。

ii. 日志文件

·报警日志文件（alert.log 或 alrt<SID>.ora）

记录数据库启动，关闭和一些重要的出错信息。数据库管理员应该经常检查这个文件，并对出现的问题作出即使的反应。你可以通过以下 SQL 找到他的路径 select value from v$parameter where name ='background_dump_dest'，或通过参数文件获得其路径。

·后台跟踪文件

路径与报警文件路径一致，记载了系统后台进程出错时写入的信息。

·用户跟踪文件

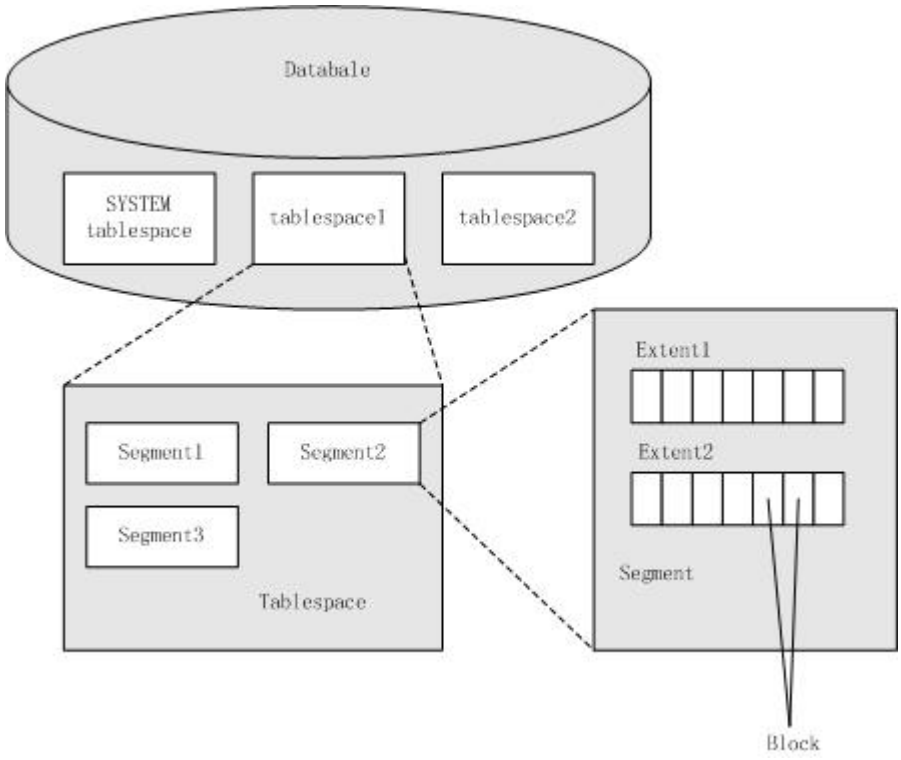
记载了用户进程出错时写入的信息，一般不可能读懂，可以通过 ORACLE 的 TKPROF 工具转化为可以读懂的格式。用户跟踪文件的路径，你可以通过以下 SQL 找到他的路径 select

value from v$parameter where name ='user_dump_dest'，或通过参数文件获得其路径。

可以通过设置用户跟踪或 dump 命令来产生用户跟踪文件，一般在调试、优化、系统分析中有很大的作用。

第四章．ORACLE 逻辑结构（Logical structure）

ORACLE 逻辑结构由一系列有相互关系的逻辑对象组成。如图：



图五 LOGICAL STRUCTURE

4.1 表空间(tablespace)

表空间是数据库中的基本逻辑结构，一系列数据文件的集合。一个表空间可以包含多个数据文件，但是一个数据文件只能属于一个表空间。

在 8i 以前，表空间的管理类型只有一种，被称为字典管理表空间（DMT），即在数据字典中管理表空间中的空间的分配。在 8i 以后的版本，为了减少在字典上的开销，引入了本地管理的表空间（LMT），在该类型的表空间中，在每个数据文件中存储的位图来管理空间的分配，不再要求使用数据字典。本地管理的表空间有速度快，无碎片等众多优点，建议用户表空间都实现本地管理。

通过 v$tablespace 可以查询表空间，DBA_TABLESPACE 可以查询详细表空间信息。

4.2 段(Segment)

段是对象在数据库中占用的空间，虽然段和数据库对象是一一对应的，但段是从数据库存储的角度来看的。一个段只能属于一个表空间，当然一个表空间可以有多个段。

表空间和数据文件是物理存储上的一对多的关系，表空间和段是逻辑存储上的一对多的关系，段不直接和数据文件发生关系。一个段可以属于多个数据文件，关于段可以指定扩展到哪个数据文件上面。

段基本可以分为以下四种

- 数据段(Data Segment)
- 索引段(Index Segment)
- 回滚段(Rollback Segment)
- 临时段(Temporary Segment)

通过 DBA/ALL/USER_SEGMENTS 可以查询详细的段信息。

4.3 区间(Extent)

关于 Extent 的翻译有多种解释，有的译作扩展，有的译作盘区，我这里通常译为区间。在一个段中可以存在多个区间，区间是为数据一次性预留的一个较大的存储空间，直到那个区间被用满，数据库会继续申请一个新的预留存储空间，即新的区间，一直到段的最大区间数(Max Extent)或没有可用的磁盘空间可以申请。

在 ORACLE8i 以上版本，理论上一个段可以无穷个区间，但是多个区间对 ORACLE 却是有性能影响的，ORACLE 建议把数据分布在尽量少的区间上，以减少 ORACLE 的管理与磁头的移动，但是在某些特殊情况下，需要把一个段分布在多个数据文件或多个设备上，适当的加多区间数也是有很大好处的。

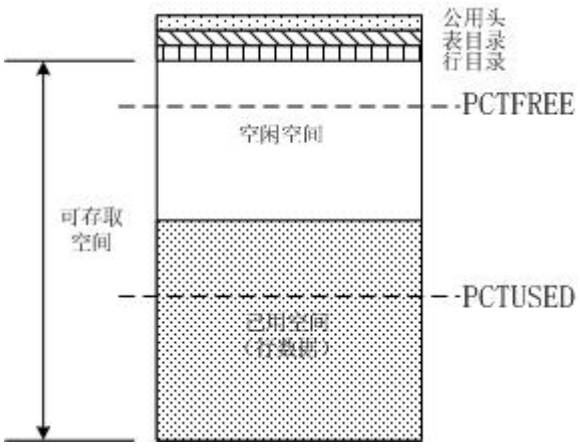
通过 DBA/ALL/USER_EXTENTS 可以查询详细的区间信息。

4.4 Oracle 数据块(Block)

ORACLE 最基本的存储单位，在建立数据库的时候指定，虽然在初始化文件中可见，但是不能修改。为了保证存取的速度，它是 OS 数据块的整数倍。ORACLE 的操作都是以块为基本单位，一个区间可以包含多个块，如果区间大小不是块大小的整数倍，ORACLE 实际也扩展到块的整数倍。

块的内部结构与数据的存取方法都是比较复杂的，以表段的块为例，从简单的结构上划分，可以把块的内部划分成如下几个部分：公用头，表目录，行目录，可存取空间等。

以下是一个表块的大致结构图：



图六 TABLE BLOCK

块头(BLOCK HEADER)包含着关于块类型（表块、索引块等等）的信息、关于块上活动和过时事务信息、磁盘上筷的地址的信息。表目录(Table directory)，如果给出的话，包含着此块中存储各行的表的信息（多个表的数据可能保存在同一个块中）。行目录（Row directory）包含在块中发现的描述行的信息。以上 3 部分为块的开销(Block Overhead)，其余部分为可用存储空间，可以用如下查询获得可用空间大小。

Select kvisval,kvistag,kvisdsc from sys.x$kvis;

一般的 8K(8192)的块可用空间为 8168

PCTFREE 与 PCTUSED 是表的两个存取参数，其实是作用在表中的块上面的，PCTFREE 与 PCTUSED 表示两个百分比，默认分别是 10 与 40。PCTFREE 表示保留该百分比的可用空间用于以后的行更新，避免行迁移。如果行数据达到 PCTFREE 保留的空间，该块从 FREE LIST 上撤消下来，不再接收数据。PCTUSED 表示当行的空闲空间降低（如删除数据）到该参数指定的百分比的时候，该块重新进入 FREE LIST，开始接收新的数据。PCTFREE 与 PCTUSED 的配置与系统的优化有一定的关系，所以要慎重，PCTFREE+PCTUSED 不要大于等于 100，否则将导致块不断的在 FREELIST 移上移下，严重影响性能。

4.5 基本表空间介绍

4.5.1 系统表空间(System)

该表空间包含的数据文件称为系统数据文件。

该存放系统表和数据字典，一般不放用户的数据，但是用户脚本，如过程，函数，包等却是保存在数据字典中的。

数据字典是一些系统表或视图，他存放系统的信息，他包括数据库版本，数据文件信息，表与索引等段信息，系统的运行状态等各种和系统有关的信息和用户脚本信息。数据库管理员可以通过对数据字典的查询，就可以了解到 Oracle 的运行状态。

查看数据数据字典的 SQL

select * from dict

查看内部系统表的 SQL

select * from v$fixed_view_definition

DBA 对系统的系统表中的数据字典必须有一个很深刻的了解，他们必须准备一些基础的 SQL 语句，通过这些 SQL 可以立即了解系统的状况和数据库的状态，这些基本的 SQL

包括

· 系统的剩余空间
· 系统的 SGA
· 状态系统的等待
· 用户的权限
· 当前的用户锁
· 缓冲区的使用状况等

在成为 DBA 的道路上我们不建议你过分的依赖于 OEM/Quest 等优秀的数据库管理工具，因为他们不利于你对数据数据字典的理解，SQL 语句可以完成几乎全部的数据库管理工作。

大量的读少量的写是该表空间的一个显著的特点。

4.5.2 临时表空间(Temporary)

该表空间包含的数据文件称为临时数据文件

主要存放用户的排序等临时数据，因为没有办法在一个永久表空间上开辟临时段，所以就必须有一个临时表空间，主要用于不能在内存上进行的排序操作。我们必须为用户指定一个临时表空间。

临时段占有的空间会在下次系统启动的时候全部被释放。

4.5.3 回滚段表空间(Rollback)

如果数据库进行对数据的修改，那么就必须使用回滚段，回滚段是用来临时存放修改前的数据(UNDO)。回滚段通常都放在一个单独的表空间上（回滚表空间），避免表空间碎片化，这个表空间包含的数据文件就是回滚数据文件。

4.5.3.1 回滚段在系统中的作用

当数据库进行更新插入删除等操作的时候，新的数据被更新到原来的数据文件，而旧的数据(Before Image)就被放到回滚段中，如果数据需要回滚，那么可以从回滚段将数据再复制到数据文件中。来完成数据的回滚。在系统恢复的时候， 回滚段可以用来回滚没有被 commit 的数据，解决系统的一直性读。

回滚段在一般情况下都是大量的写，少量读，因此建议把回滚段单独出来放在一个单独的设备（如单独的磁盘或 RAID），以减少磁盘的 IO 争用。

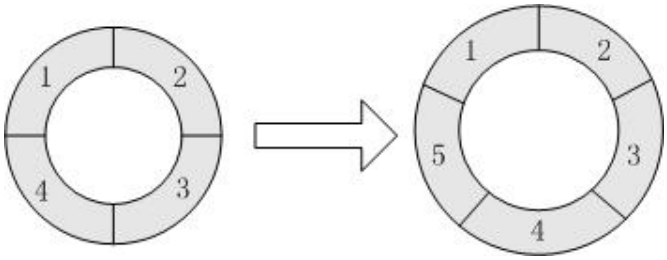
4.5.3.2 回滚段的工作方式

· 一个回滚表空间可以被划分成多个回滚段.
· 一个回滚段可以保存多个会话的数据.
· 回滚段是一个圆形的数据模型

假设回滚段由 4 个区间组成，他们的使用顺序就是区间 1→区间 2→区间 3→区间 4→区间 1。也就是说，区间是可以循环使用的，当区间 4 到区间 1 的时候，区间 1 里面的会话

还没有结束，区间 4 用完后就不能再用区间 1,这时系统必须分配区间 5，来继续为其他会话服务服务。这也就是为什么回滚段在大事务的情况下会不断“ 涨大 ”的原因，回滚段的扩充是影响性能的，要尽量避免。

这是一个回滚段从 4 个区间简单的扩充到 5 个区间的例子：



图七 ROLLBACK SEGMENT

我们分析一个 Update 语句的完成

1、用户提交一个 Update 语句

2、Server Process 检查内存缓冲.

如果没有该数据块的缓冲，则从磁盘读入

i. 如果没有内存的有效空间，DBWR 被启动将未写入磁盘的脏缓冲写入磁盘

ii. 如果有有效空间，则读入

3、在缓冲内更新数据

i. 申请一个回滚段入口，将旧数据写如回滚段

ii. 加锁并更新数据

iii. 并在同时将修改记录在 Redo log buffer 中

4、用户提交一个 Commit 语句

i. SCN 增加

ii. 将 Redo log buffer 写入 Redo log file

iii. 返回用户 Commit 完成

4.5.4 用户表空间(User)

其包含的数据文件称为用户数据文件

一般是由用户建立，来存取用户数据的表空间，一般有两类常见的用户型数据，数据和索引，一般来说，如果条件许可的话，可以考虑放在不同的磁盘上。

第五章. 常见问题

1、实例和 SID 的关系是什么？

经常有人问 SID 是什么？在 Oracle 系统中 SID 是一个经常出现的变量，如环境变量 ORACLE_SID， 初始化文件 initSID.ora，那究竟什么是 SID 呢？其实 SID 就是 Oracle 实例的标识，不同的 SID 对应不同的内存缓冲(SGA)和不同的后台进程。这样一来我们就可以得当在一台物理的服务器上可以有多个 SID 的数据库实例。

2、Oracle 数据库和实例的关系是什么？

数据库是由物理文件和存取数据文件的实例组成，当存取数据文件的实例是一个的时

候，数据库被称做单节点数据库。这是我们看到的最多的数据库形式。当然还有一种多节点数据库，就是一个以上的实例共同访问一个数据库(或者说共同访问一组数据文件)， 更好的提供稳定性和并行处理能力。这在 8i 中被称为 OPS(Oracle Parallel Server)，在 Oracle9i 中被称为 RAC(real application cluster)。在这种数据库中。两个/多个实例分别在不同服务器上，所有 Oracle 数据文件在共享的磁盘阵列上，多个服务器上的实例可以同时工作，他们通过一个内部的网络进行通信。如果一台服务器不能提供服务的话，另一台会接管它的工作，特别是在关键的业务有很大的潜力。

3、在运行的数据库中数据文件中是不是可能存在没有被提交的数据？

这是可能存在的，因为用户数据文件的数据是由 DBWR 写入的，DBWR 是一个很底层的后台进程，不负责与用户交互。用户的交互是由 LGWR 完成的。

4、在问题 3 中，如果存在没有写入的数据，那么机器突然断电，数据完整性会不会损坏？

不会的，因为数据库的完整性是 LGWR 来保证的，而且 ORACLE 保证了 DBWR 写入数据文件的任何修改已经被记录在重做日志文件中。当系统再次启动的时候，通过读取重做日志文件就可以知道那些数据没有被提交。这时候 ORACLE 会自动回滚那些数据。所以说联机日志的损坏，特别是当前联机日志的损坏，对数据库的影响是巨大的，可能会导致数据库的不完整。

5、数据文件损坏会丢失数据吗？

可以这么说，如果你有备份和归档，就不会。因为所有对数据修改的记录都在重做日志中有记录，所以不会丢失数据，你只要恢复以前的备份再用归档日志文件恢复和当前的在线重做日志就可以恢复所有数据。

6、在线重做日志损坏会丢失数据吗？

以上说了，在线日志对数据库的损坏是极大的，所以不仅可能丢失数据，还可能引起数据库的不同步。在重做日志中的所有 commit 的记录都会丢失，这也是 Oracle 为什么要对在线重做日志文件做镜像的原因。任何的数据丢失都是不允许的。

7、我在事务能不能指定不写回滚段？

不可以的，写回滚段是 ORACLE 保证一致性读和事务一致性的根本。回滚段是高写入段，建议把它放到单独的设备上来。

对于 DDL 语句，如 DROP,TRUNCATE 却可以不写回滚段（没有 UNDO 信息），所以对于整个表的删除，如果数据量比较大，建议用 Truncate Table 的方法。

不写联机日志也是不可能的，但可以在某些特定操作中，可以写很少的联机日志，如以 NOLOGGING 的方式通过 Create table tablename as select 创建表，或以 Append 的方式 Insert 数据到表，或直接载入等操作。

第六章. 小结

这里，我们了解了实例和数据库的关系，一个数据库可以有多个实例，但是一个实例却不可能对应多个数据库，在一般的情况下，我们都是用的单节点数据库，即一个实例仅仅对应一个数据库。

我们了解了 ORACLE 实例的组成，包括内存和后台进程，进一步解释了 SGA 的组成与 SGA 的作用，并分析了语句重用的好处。在后台进程中，重要的阐述了 DBWR 与 LGWR，其中 DBWR 是一个底层的由 ORACLE 控制的后台进程，而 LGWR 负责与用户交互.

在 ORACLE 数据库中,我们重要阐述了数据库的物理与逻辑结构，在物理结构中，需要注意四类以下文件：控制文件，联机日志，数据文件与参数文件。在逻辑结构中，需要清楚

每个逻辑结构的关系，从大到小的顺序为：表空间→段→区间→块。

第二部分：ORACLE9i 新特性

一、创建数据库

9i 创建数据库的工具改名为 DBCA，也可以通过脚本更轻松容易的创建数据库，如：

```
create database ${ORACLE_SID}
    user sys identified by sys
    user system identified by system
    logfile group 1 ('${ORACLE_BASE}/oradata/${ORACLE_SID}/redo01.log') size 10M,
    group 2 ('${ORACLE_BASE}/oradata/${ORACLE_SID}/redo02.log') size 10M,
    group 3 ('${ORACLE_BASE}/oradata/${ORACLE_SID}/redo03.log') size 10M
    maxlogfiles 5
    maxlogmembers 5
    maxloghistory 1
    maxdatafiles 254
    maxinstances 1
    archivelog
    character set ZHS16GBK
    national character set AL16UTF16
    datafile '${ORACLE_BASE}/oradata/${ORACLE_SID}/system01.dbf' size 300M
    default temporary tablespace tbstemp tempfile
            '${ORACLE_BASE}/oradata/${ORACLE_SID}/temp01.dbf' size 500M
    undo tablespace tbsundo datafile
            '${ORACLE_BASE}/oradata/${ORACLE_SID}/undo01.dbf' size 500M;
```

其特点为使用专用的回滚和临时表空间，而不象 Oracle 8i 中的那样，回滚和临时表空间与普通表空间没有差异，这样既简化了配置也有利于效能提高。要注意临时表空间的指定文件关键字是 tempfile 而不是通用的 datafile，而且临时表空间的存储选项由 Oracle 系统决定。同样回滚表空间也是由 Oracle 系统决定，不必人工干预。

二、二进制的参数文件

Oracle 9i 在$ORACLE_HOME/dbs 下可使用二进制配置文件，缺省为 spfile{SID}.ora，如 spfileoradb.ora，支持 Oracle 系统进程在不重启的情况下动态调整参数，这对要求不间断运行的系统是有利的。在建库阶段就可将此配置文件创建起来。

create spfile from pfile;

or

create spfile='文件全名' from pfile='文件全名'

数据库在启动的时候，会默认在$ORACLE_HOME/dbs 找 spfile{实例名}.ora 文件，如果没有，它将寻找 pfile 文件，就是与 8i 兼容的 init{SID}.ora 的文本文件。当然，也可以通过指定的方式启动数据库。

Startup pfile='文件全名' or startup spfile='文件全名'

使用 spfile 最大的好处可能就是支持动态的参数修改，如

alter system set parameter_name=new_value scope=memory|spfile|both

这样，通过命令方式可以修改参数生效或到参数文件。对于 24*7 的系统来说，这是一个不小的好处。

三、SGA 动态内存分配

ORACLE9i 中可以动态分配 SGA，但是也是有限制的，你可以但分配 SGA_MAX_SIZE 大小，在这个尺寸之内，SGA 的大小是可以动态分配的，可以根据不同的使用要求，来分配不同的 SGA 内存大小。

如

```
SQL> show parameter sga_max_size

NAME                                 TYPE        VALUE
------------------------------------ ----------- --------------
sga_max_size                         big integer 1605044320

SQL> show parameter db_cache_size

NAME                                 TYPE        VALUE
------------------------------------ ----------- ---------
db_cache_size                        big integer 671088640

SQL> alter system set db_cache_size=500M scope=both;
System altered.
```

注：db_cache_size 也是 9i 新增加的参数，代替了以前版本中的 db_block_buffers，可以通过该参数直接设置数据缓冲区的大小。

四、PGA 自动管理

在 9i 以前，PGA 主要包括用户信息，session 信息，堆栈信息和 Sort_Area_Size, Hash_Area_Size, Bitmap_Merge_Size, Create_Bitmap_Area_Size 等池的大小。我们必须设置以上池的大小并监控它以获得最佳性能。

但是，在 9i 以后，我们可以通过设置以下两个参数以获得 PGA 的自动管理：

a. WORKAREA_SIZE_POLICY

b. PGA_AGGREGATE_TARGET

如果 WORKAREA_SIZE_POLICY=true 和 PGA_AGGREGATE_TARGET=具体内存大小，ORACLE 将自动管理 PGA 大大小，而忽略以上参数的设置。自动管理的 PGA 将采用自己的管理方法，可以获得 PGA 的共享使用。

如果采用了自动管理，将有如下视图可以监控

1. V$SQL_WORKAREA
2. V$SQL_WORKAREA_ACTIVE
3. V$PROCESS contains new columns (PGA_USED_MEM, PGA_ALLOC_MEM AND PGA_MAX_MEM)
4. V$PGASTAT

Oracle 9.2 又增加了如下视图

1. V$SQL_WORKAREA_HISTOGRAM
2. V$PGA_TARGET_ADVICE
3. V$PGA_TARGET_ADVICE_HISTOGRAM

五、表空间管理的本地化

在 9i，特别是 920 以后，如果系统表空间采用了本地管理，那所有的表空间都只能采用本地管理模式，如果显示指定字典管理将显示错误。其具体语法可以简单的描述为：

create tablespace tbsdata datafile '…' [extent management local] [autoallocate];

而对于指定每个扩展块大小的创建策略，设立了新选项：统一扩展块大小（uniform [size xxx[K|M]]），可覆盖 autoallocate 选项，如果不加上具体的 size xxx[K|M]，缺省为 1M，这样就不必考虑 Oracle 8i 中的如 initial,next,pctincrease,maxextents 等 default storage 参数应如何组合，事实上 Oracle 8i 的这些设置原本就没有什么意义。

如果未指定 extent management 的类型，如果又同时使用 default storage 选项，为了兼容 ORACLE8i，就有以下的判断：

如果使用 minimun extent，Oracle 检查是否 minumum extent=initial=next 且 pctincrease＝0,如是，Oracle 使用 uniform 选项，size=initial 如不是，Oracle 忽略指定选项，使用 autoallocate。

如果未指定 minimum extent，Oracle 检查是否 initial=next 且 pctincrease＝0,如是 Oracle 使用 uniform 选项，size=initial；如不是 Oracle 忽略指定选项，使用 autoallocate。

为了避免与 Oracle 8i 的习惯做法混淆，建议只使用 Oracle 9i 较简洁的方法。

对于存储少量静态数据的表空间来说，如配置信息等，可简单地写为：

create tablespace tbsdata datafile '…';

对于必须关心其扩展块大小的表空间，如大批量的记录或索引，可简单地写为：

create tablespace tbsdata datafile '…' uniform size 10M;

另外一个问题就是，9i 的临时表空间将全部采用本地管理模式，所以在 ORACLE8i 往

上升级的时候，临时表空间将会出错，因为 ORACLE 这里没有提供向下的兼容性。

创建临时表空间的语法在 9i 中被改为如下：

create temporary tablespace tempfile ‘文件全名’

六、自动回滚段管理

在 9i 以前，回滚段全是手工管理与监控的，DBA 需要花费一定的时间去管理与监控回滚段的性能，创建不好或管理不好的回滚段，将引起很大的性能瓶颈。从 ORACLE9i，为了更好的管理回滚段，ORACLE，默认采用自动回滚段管理。

自动回滚段管理可以最大限度的避免 8i 中比较有名的“ 快照太老 ”的错误，ORACLE9i 通过如下四个初试化参数来设置自动回滚段管理：

undo_management	string	AUTO
undo_retention	integer	10800
undo_suppress_errors	boolean	FALSE
undo_tablespace	string	UNDOTBS1

undo_management 表明了回滚段管理采用自动方式，ORACLE 建议采用自动方式，如果不是对数据库非常了解，不要修改该参数。

undo_retention 表明了回滚信息在回滚段中保持的时间，单位是秒，默认 3 个小时。

undo_suppress_errors 表明不显示某些错误信息，如对系统回滚段的操作将不显示错误，虽然这个操作没有成功。

undo_tablespace 表明了使用自动回滚的表空间，DBA 需要监控该表空间的大小。

自动回滚段的另外一个好处就是可以利用 flashback 来查看提交以前的数据或导出提交以前的数据，防止一定程度上的人为错误。

七、自动段空间分配

ORACLE9i 引进了段空间的自动分配（ASSM），ASSM 的 tablespace 是通过将 SEGMENT SPACE MANAGEMENT AUTO 子句添加到 tablespace 的定义句法里而实现的（只能是本地管理的表空间）。通过使用位图 freelist 取代传统单向的链接列表 freelist，ASSM 的 tablespace 会将 freelist 的管理自动化，并取消为独立的表格和索引指定 PCTUSED、FREELISTS 和 FREELIST GROUPS 存储参数的能力。

要注意，当表或者索引被分配到这个 tablespace 以后，用于独立对象的 PCTUSED 的值会被忽略，而 Oracle9*i* 会使用位图数组来自动地管理 tablespace 里表格和索引的 freelist。对于在 LMT 的 tablespace 内部创建的表格和索引而言，这个 NEXT 扩展子句是过时的，因为由本地管理的 tablespace 会管理它们。但是，INITIAL 参数仍然是需要的，因为 Oracle 不可能提前知道初始表格加载的大小。对于 ASSM 而言，INITIAL 最小的值是三个区块。

关于一个万能的方法对于 Oracle 来说是否是最好的方法还有一些争论。在大型数据库里，单独的对象设置会带来性能和存储上的巨大不同。

在以前的版本中，在没有多个 freelist 的时候，每个 Oracle 表格和索引在表格的头部都曾有一个数据块，用来管理对象所使用的剩余区块，并为任何 SQL 插入声明所创建的新数据行提供数据块。当数据缓冲内的数据块由于被另一个 DML 事务处理锁定而无法使用的时候，缓冲区忙等待就会发生。当你需要将多个任务插入到同一个表格里的时候，这些任务就被强制等待，而同时 Oracle 会在同时分派剩余的区块，一次一个。

有了 ASSM 之后，Oracle 宣称显著地提高了 DML 并发操作的性能，因为（同一个）位图的不同部分可以被同时使用，这样就消除了寻找剩余空间的串行化。根据 Oracle 的测试结果，使用位图 freelist 会消除所有分段头部（对资源）的争夺，还能获得超快的并发插入操作

参考文献：

http://www.happyit.net

http://otn.oracle.com/documentation/content.html

http://metalink.oracle.com/

Expert One-on-one Oracle [美] Thomas Kyte 著 清华大学出版社

Oracle 8i Web 开发指南 [美] Dan Hotka,et al. 著 清华大学出版社

Oracle 8i DBA Architecture & Administration and backup & Recovery Study Guide [美] Dong Stuns Biju Thomas 著 电子工业出版社

Oracle 数据库管理员技术指南 [美] Sumit Sarin 著 机械工业出版社

ORACLE 9i UNIX 管理手册 [美] Donald K.Burleson 著 机械工业出版社