第3章

Oracle数据库

PPT 视频讲解

3.1 Oracle 数据库概述

在本书第 1 章中曾指出，Oracle 服务器是由静态的数据库和动态的 Oracle 实例组成的。本章将介绍 Oracle 数据库。

数据库是数据存储的容器，用来收集、存储数据和返回信息。Oracle 数据库可以从逻辑存储结构和物理存储结构两个角度来划分层次。

物理存储结构主要用于描述在 Oracle 数据库外部数据的存储，即在操作系统层面中如何组织和管理数据，与具体的操作系统有关；逻辑存储结构主要描述 Oracle 数据库内部数据的组织和管理方式，即在数据库管理系统的层面中如何组织和管理数据，与操作系统没有关系。物理存储结构具体表现为一系列的操作系统文件，是可见的；逻辑存储结构是物理存储结构的抽象体现，是不可见的，可以通过查询数据库数据字典来了解逻辑存储结构。

Oracle 数据库的物理存储结构与逻辑存储结构既相互独立又相互联系，它们之间的关系如图 3-1 所示。

从图 3-1 中可以看出，Oracle 数据库的物理存储结构和逻辑存储结构的基本关系可以归纳如下所述。

(1) 一个数据库在物理上包含多个数据文件，在逻辑上包含多个表空间。

(2) 一个表空间包含一个或多个数据文件，一个数据文件只能从属于某个表空间。

(3) 一个区只能从属于一个数据文件，而一个数据文件可包括一个或多个区。

(4) Oracle 数据库的块由一个或多个操作系统块组成。

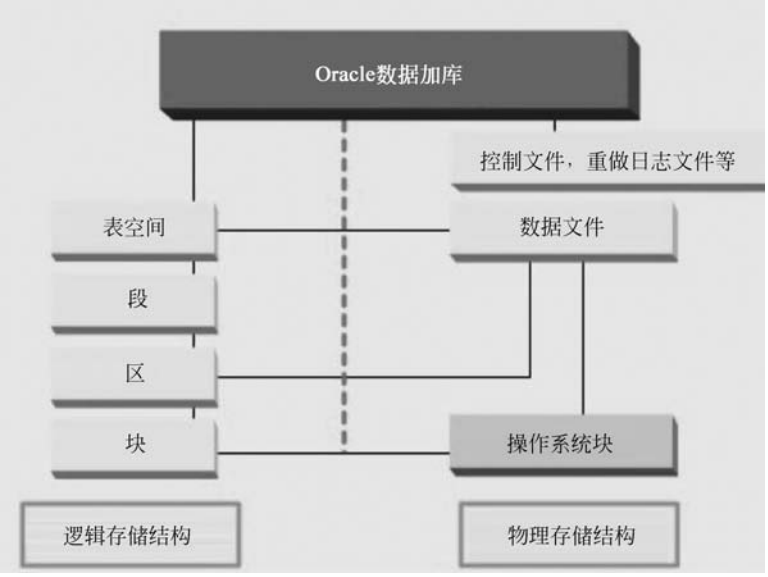

图 3-1 Oracle 数据库的物理存储结构与逻辑存储结构的关系示意

3.2 Oracle 数据库物理存储

Oracle 数据库物理存储是存储在磁盘中的操作系统文件，它的组成如图 3-2 所示。

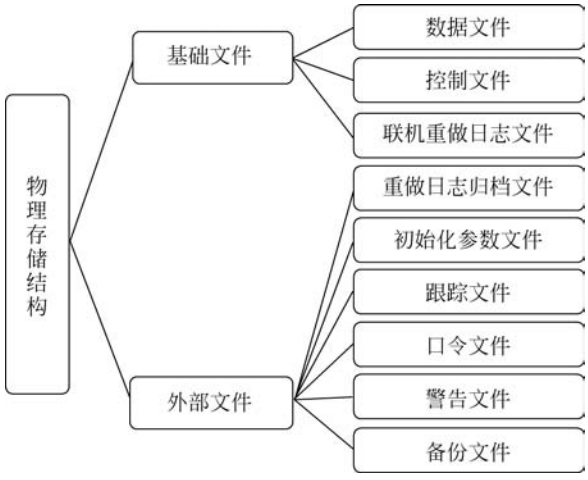

图 3-2 Oracle 数据库物理存储的组成

其中最主要的 3 种类型文件分别为数据文件、控制文件和联机重做日志文件，这三大核心文件对 Oracle 数据库的正常启动是缺一不可的。具体的物理存储结构组成文件如下。

(1) 数据文件：用于存储数据库中的所有数据。

(2) 控制文件：用于记录和描述数据库的物理存储结构信息。

(3) 联机重做日志文件：用于记录外部程序(用户)对数据库的改变操作。

(4) 重做日志归档文件:用于保存已经写满的重做日志文件。

(5) 初始化参数文件:用于设置数据库启动时的参数初始值。

(6) 跟踪文件:用于记录用户进程、数据库后台进程等的运行情况。

(7) 口令文件:用于保存具有 sysdba、sysoper 权限的用户名和用户口令。

(8) 警告文件:用于记录数据库的重要活动以及发生的错误。

(9) 备份文件:用于存放数据库备份所产生的文件。

3.2.1 数据文件

数据文件就是用来存放数据库数据的物理文件,文件后缀为".dbf"。数据文件存放的主要内容可以归纳为表中的数据,索引数据,数据字典定义,回滚事务所需信息,存储过程、函数和数据包的代码,以及用来排序的临时数据。

数据文件可以通过动态性能视图进行查看,也可以进行很多操作。关于数据文件的具体操作将在下面具体阐述。

1. 数据文件相关视图

若要了解数据文件中的具体信息,首先需要熟悉数据文件的相关视图,具体如下所述。

(1) dba_data_files:包含数据库中所有数据文件的信息,包括数据文件所属的表空间、数据文件编号等。

(2) dba_temp_files:包含数据库中所有临时数据文件的信息。

(3) dba_extents:包含所有表空间中已分配的区的描述信息。

(4) user_extents:包含当前用户所拥有的对象在所有表空间中已分配的区的描述信息。

(5) dba_free_space:包含表空间中空闲区的描述信息。

(6) user_free_space:包含当前用户可访问的表空间中空闲区的信息。

(7) v$datafile:包含从控制文件中获取的数据文件信息。

(8) v$datafile_header:包含从数据文件头部获取的信息。

(9) v$tempfile:包含所有临时文件的基本信息。

2. 查询看数据文件信息

(1) 查询数据文件的动态信息,示例如下:

```
SQL> col name for a50
SQL> select file#,name from v$datafile;

    FILE#   NAME
--------- --------------------------------
    1       D:\ORACLE19C\ORADATA\NEWORCL\SYSTEM01.DBF
    3       D:\ORACLE19C\ORADATA\NEWORCL\SYSAUX01.DBF
    4       D:\ORACLE19C\ORADATA\NEWORCL\UNDOTBS01.DBF
    7       D:\ORACLE19C\ORADATA\NEWORCL\USERS01.DBF
```

（2）查询数据文件的详细信息，示例如下：

```
SQL> col file_name for a50
SQL> col tablespace_name for a20
SQL> select file_id,file_name,tablespace_name,bytes/(1024 * 1024)MB
  2  from dba_data_files order by tablespace_name;

FILE_ID  FILE_NAME                                        TABLESPACE_NAME  MB
-------  ----------------------------------------------  ---------------  -------
3        D:\ORACLE19C\ORADATA\NEWORCL\SYS  AUX01.DBF SYS  AUX              720
1        D:\ORACLE19C\ORADATA\NEWORCL\SYSTEM01.DBF        SYSTEM           910
4        D:\ORACLE19C\ORADATA\NEWORCL\UNDOTBS01.DBF       UNDOTBS1         65
7        D:\ORACLE19C\ORADATA\NEWORCL\USERS01.DBF         USERS            5
```

（3）查询数据文件的增长方式，示例如下：

```
SQL > SELECT TABLESPACE_NAME,BYTES, AUTOEXTENSIBLE, FILE_NAME
  2 FROM DBA_DATA_FILES;

TABLESPACE_NAME  BYTES        AUTOEXTENSIBLE  FILE_NAME
---------------  ----------  --------------  ----------------------------------
SYSTEM           954204160    YES             D:\ORACLE19C\ORADATA\NEWORCL\SYSTEM01.DBF
SYSAUX           754974720    YES             D:\ORACLE19C\ORADATA\NEWORCL\SYSAUX01.DBF
USERS            5242880      YES             D:\ORACLE19C\ORADATA\NEWORCL\USERS01.DBF
UNDOTBS1         68157440     YES             D:\ORACLE19C\ORADATA\NEWORCL\UNDOTBS01.DBF
```

（4）查询临时数据文件信息，示例如下：

```
SQL> select tablespace_name, file_name,
  2  autoextensible from dba_temp_files;

TABLESPACE_NAME      FILE_NAME                                          AUT
---------------  ------------------------------------------------  -----
TEMP                D:\ORACLE19C\ORADATA\NEWORCL\TEMP01.DBF            YES
```

3. 重置数据文件大小

（1）观察每个数据文件的空间大小，示例如下：

```
SQL> select tablespace_name, file_id, file_name, round (bytes /(1024 * 1024 ),0 ) total_space
  from dba_data_files order by tablespace_name;

TABLE       SPACE_NAME  FILE_IDFILE_NAME                      TOTAL_SPACE
--------  ----------  ------------------------------  ----------
```

```
SYSAUX      3          D:\ORACLE19C\ORADATA\NEWORCL\SYSAUX01.DBF    720
SYSTEM      1          D:\ORACLE19C\ORADATA\NEWORCL\SYSTEM01.DBF    910
UNDOTBS1    4          D:\ORACLE19C\ORADATA\NEWORCL\UNDOTBS01.DBF   65
USERS       7          D:\ORACLE19C\ORADATA\NEWORCL\USERS01.DBF     5
```

(2) 观察每个数据文件的使用情况,示例如下:

```
SQL> select file_name, a.file_id, sum (a.bytes )/1024 /1024 as MB
  2  from dba_extents a, dba_data_files b where a.file_id = b.file_id
  3  group by file_name, a.file_id;

FILE_NAME                                          FILE_ID         MB
-------------------------------------------------- ------- ----------
D:\ORACLE19C\ORADATA\NEWORCL\USERS01.DBF                 7     1.6875
D:\ORACLE19C\ORADATA\NEWORCL\SYSTEM01.DBF                1   900.6875
D:\ORACLE19C\ORADATA\NEWORCL\SYSAUX01.DBF                3     645.75
D:\ORACLE19C\ORADATA\NEWORCL\UNDOTBS01.DBF               4     32.125
```

(3) 重置文件大小,示例如下:

```
SQL> alter database datafile 'D:\ORACLE19C\ORADATA\NEWORCL\SYSAUX01.DBF' resize 750MB;

数据库已更改.
```

4. 移动数据文件

若某个磁盘的I/O操作过于繁忙,就可能影响到Oracle数据库系统的整体效率,此时需将一个或几个数据文件进行移动。若某个磁盘已经毁损,为了能使数据库系统继续运行,要将一个或多个数据文件移动。移动数据文件的操作语句有如下两种。

(1) alter tabespace 表空间名 rename datafile 文件名 to 文件名。

该语句只适用于上面没有活动的还原数据或临时段的非系统表空间中的数据文件。要求在使用该语句时,表空间必须为脱机状态且目标数据文件必须存在,因为该语句只修改控制文件中指向数据文件的指针(地址)。

(2) alter database 数据库名 rename file 文件名 to 文件名。

该语句适用于系统表空间和不能置为脱机的表空间中的数据文件。要求在使用该语句时,数据库必须运行在加载(Mount)状态且目标数据文件必须存在,因为该语句只修改控制文件中指向数据文件的指针(地址)。

3.2.2 控制文件

控制文件用于记录和维护数据库的全局物理结构,它就像数据库的一个管家,是成功启动和操作数据库必须的二进制文件,以".ctl"为文件后缀。一个数据库至少需要一个控制文件,每个控制文件只与一个数据库相关联。

控制文件中包含的记录项有数据库名称和标识符、数据库创建时间、表空间名称、数据文件和联机重做日志的名字和位置、当前联机重做日志序号、检查点信息、还原段的开始与结束、重做日志归档/存档信息和备份信息。

1. 控制文件相关视图

若要进行控制文件的有关操作，首先需要了解有关控制文件的视图，如下所示。

(1) v$controlfile：列出所有与当前 Oracle 实例相关的控制文件的名和状态。

(2) v$parameter：列出所有参数的状态和位置。

(3) v$controlfile_record_section：给出控制文件记录段相关的信息。

2. 获取控制文件

通过获取控制文件，可以发现每个数据库通常包含两个或更多控制文件，这几个控制文件在内容上保持一致，也分配在相同的物理硬盘中。但当数据库或硬盘损坏时，可利用备份的控制文件启动 Oracle 实例，以提高数据库的可靠性。

```
SQL> select name from v$controlfile;
NAME
--------------------------------------------------------------------------------
D:\ORACLE19C\ORADATA\NEWORCL\CONTROL01.CTL
D:\ORACLE19C\ORADATA\NEWORCL\CONTROL02.CTL
```

或者

```
SQL> select value from v$parameter where name = 'control_files';

VALUE
--------------------------------------------------------------------------------
D:\ORACLE19C\ORADATA\NEWORCL\CONTROL01.CTL, D:\ORACLE19C\ORADATA\NEWORCL\CONTROL02.CTL
```

3. 备份控制文件

控制文件是一个极其重要的文件，除了将控制文件的多个副本存在不同的硬盘上这个保护措施外，在数据库的结构发生变化之后，还应该立即对控制文件进行备份。

```
SQL> alter database backup controlfile to 'D:\BACKUP\CONTROL.BAK';

数据库已更改.
```

或者

```
SQL> alter database backup controlfile to trace;

数据库已更改.
```

创建控制文件的命令并将其备份到一个跟踪文件中，获得当前跟踪文件生成路径。

```
SQL> select tracefile from v$process where addr in (select paddr from v$session where sid in
(select sid from v$mystat));

TRACEFILE
------------------------------------------------------------------
D:\ORACLE19C\diag\rdbms\neworcl\neworcl\trace\neworcl_ora_35488.trc
```

4. 添加和移动控制文件

如果控制文件在同一目录下,就需要进行移动或添加。

步骤 1　首先利用数据字典 v$controlfile 获取现有控制文件的名字。

```
SQL> select name from v$controlfile;

NAME
----------------------------------------
D:\ORACLE19C\ORADATA\NEWORCL\CONTROL01.CTL
D:\ORACLE19C\ORADATA\NEWORCL\CONTROL02.CTL
```

步骤 2　在数据库启动状态下,修改服务器端初始化参数文件 spfile,使用 alter system set control_files 命令改变控制文件的位置和新增控制文件。

```
SQL> alter system set control_files =
'D:\ORACLE19C\ORADATA\NEWORCL\CONTROL01.CTL','D:\ORACLE19C\ORADATA\NEWORCL\CONTROL02.CTL',
'D:\BACKUP\CONTROL03.CTL' SCOPE = SPFILE;

系统已更改.
```

步骤 3　关闭数据库,以确保复制后的控制文件与源控制文件内容完全相同。

```
SQL> shutdown immediate;
数据库已经关闭.
已经卸载数据库.
ORACLE 例程已经关闭.
```

步骤 4　移动或复制控制文件。可以使用当前的两个控制文件中的任意一个,复制为新的控制文件。

```
SQL> host copy D:\ORACLE19C\ORADATA\NEWORCL\CONTROL02.CTL
D:\BACKUP\CONTROL03.CTL
已复制         1 个文件.
```

步骤 5 启动数据库。

```
SQL> startup
ORACLE 例程已经启动.

Total System Global Area  2550136752 bytes
Fixed Size                   9031600 bytes
Variable Size              570425344 bytes
Database Buffers          1946157056 bytes
Redo Buffers                24522752 bytes
数据库装载完毕.
数据库已经打开.
```

步骤 6 查询是否已经成功移动到所要移动的目录下。

```
SQL>select name from v$controlfile;

NAME
----------------------------------------
D:\ORACLE19C\ORADATA\NEWORCL\CONTROL01.CTL
D:\ORACLE19C\ORADATA\NEWORCL\CONTROL02.CTL
D:\BACKUP\CONTROL03.CTL
```

3.2.3 日志文件

日志文件用来记录数据库的事务处理过程，主要保存了用户对数据库所做的更新操作，包含的主要信息是记录事务的开始和结束、事务中每项操作的对象和类型、更新操作前后的数据值等。用户对数据库所做的修改都是在数据库的高速缓冲区中进行的，同时将产生的重做记录写入重新日志缓冲区，在一定条件下由 DBWn 进程将修改后的结果成批地写回到数据文件中，而重做日志缓冲区中的重做记录由 LGWR 进程周期性地写入重做日志文件中。因为所有的处理都记录在重做日志里，所以数据库系统可以使用这些事务记录进行恢复操作，后缀名为“.log”。

为了保证 LGWR 进程的正常进行，通常采用重做日志组(Group)，每个组中包含若干完全相同的重做日志文件成员，这些成员文件互为镜像。每组内的日志文件的内容完全相同，且保存在不同的位置。它们用于磁盘日志镜像，以做多次备份提高安全性。在默认情况下，多组通常只有一组处于活动状态。采用循环写的方式进行工作。当一个重做日志写满后，LGWR 进程就会移到下一个日志组，称之为日志切换，同时信息回写到控制文件中。

重做日志结构及工作原理如图 3-3 所示。

1. 重做日志相关视图

Oracle 提供了 v$log、v$logfile 和 v$log_history 3 个视图用于维护在线重做日志。这 3

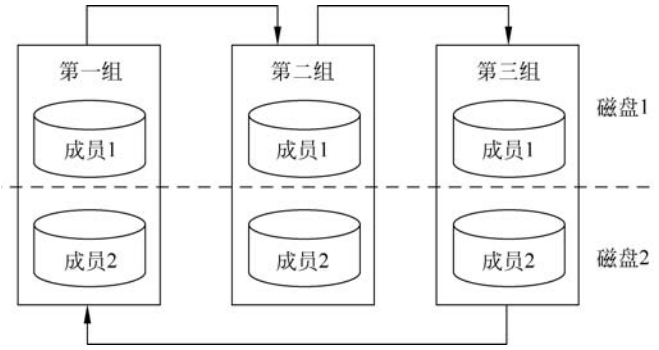

图 3-3　重做日志结构及工作原理

个视图主要用于查看和修改在线日志。

(1) v$log：包含从控制文件中获取的所有重做日志文件组的基本信息。

(2) v$logfile：包含重做日志文件组及其成员文件的信息。

(3) v$log_history：包含关于重做日志文件的历史信息。

2. 重做日志以及日志组信息查询

(1) 查询重做日志文件组的信息。

```
SQL> select group#,sequence#,members,archived,status,first_time from v$log;

    GROUP#   SEQUENCE#     MEMBERS  ARC  STATUS           FIRST_TIME
---------- --------- --------- ------ ------ -----------------
    1        1             1        YES  INACTIVE         09-2月 -20
    2        2             1        YES  INACTIVE         12-2月 -20
    3        3             1        NO   CURRENT          15-2月 -20
```

上述结果中 STATUS 代表日志组状态，它的值有以下 6 种。

① UNUSED：从未对联机重做日志文件组进行写入。这代表是刚添加的联机重做日志文件的状态。

② CURRENT：当前的联机重做日志文件组。这说明该联机重做日志文件组是活动的。

③ ACTIVE：联机重做日志文件组是活动的，但是并非当前联机重做日志文件组。数据库的崩溃恢复需要该组。它可能已归档，也可能未归档。

④ CLEARING：在执行 alter database clear logfile 命令后正在将该日志重建为一个空日志。日志清除后，其状态更改为 UNUSED。

⑤ CLEARING_CURRENT：正在清除当前日志文件中的已关闭线程。如果在切换时发生某些故障，如写入新日志标头时发生了输入/输出错误，那么日志可能处于此状态。

⑥ INACTIVE：例程恢复不再需要联机重做文件日志组。它可能已归档，也可能未归档。

(2) 查询重做日志文件的信息。

```
SQL>select * from v$logfile;

GROUP#  STATUS TYPE      MEMBER                                    IS_    CON_ID
------ ---------- ---------------------------------------- ------ -------
3            ONLINE      D:\ORACLE19C\ORADATA\NEWORCL\REDO03.LOG    NO     0
2            ONLINE      D:\ORACLE19C\ORADATA\NEWORCL\REDO02.LOG    NO     0
1            ONLINE      D:\ORACLE19C\ORADATA\NEWORCL\REDO01.LOG    NO     0
```

上述结果中 STATUS 代表日志状态,它的值可以为下列值之一。

① INVALID:表明该文件不可访问。

② STALE:表示文件内容不完全。

③ DELETED:表明该文件已不再使用。

④ 空白表明文件正在使用中。

3. 强制产生日志切换

LGWR 按顺序向联机重做日志组写入重做信息。一旦当前联机重做日志组被写满,LGWR 就开始写入下一个组,称之为日志切换(Log Switch)。通常,只有当前的重做日志组在写满后才发生日志切换,并可通过设置参数 archive_log_target 来控制日志切换的时间间隔,在必要时也可以采用手工强制进行日志切换。

(1) 设置 fast_start_mttr_target:可以在初始化参数文件中设置此参数,代表实例恢复所用时间,单位为 s。例如,设置 fast_start_mttr_target=300,代表如果数据库需要实例恢复,那么恢复的时间不超过 300s,系统会根据 300s 时间自动计算可以保留的脏数据块的数目;如果恢复的时间超过 300s,那么实例会自动发出检查点。

(2) 强制产生日志切换命令。

```
SQL>alter system switch logfile;

系统已更改.
```

(3) alter system checkpoint 命令:检查点是一个数据库事件,它用于同步所有数据文件、控制文件以及重做日志文件。在必要时,DBA 也可以手动发出检查点命令,命令如下:

```
SQL>alter system checkpoint;

系统已更改.
```

4. 重做日志的管理

通常,DBA 会在创建数据库时按照计划创建所需重做日志组和各个组成员日志文件。然而在有些情况下,需要通过手工方式为数据库添加新重做日志组和成员。例如,如果当前某个重做日志组由于某种原因无法使用,DBA 需要创建一个新的重做日志组代替它进行工作。在另外一些情况下,DBA 可能需要改变现有重做日志文件的名称和位置或删除重做日

志组或成员。

(1) 创建重做日志组语法：

```
alter dtabase add logfile [group n]
```

(2) 删除重做日志组语法：

```
alter database drop logfile [group n]
```

(3) 创建重做日志文件组成员文件：

```
alter database add logfile member '文件名'to group n
```

(4) 删除重做日志文件组成员文件：

```
alter database drop logfile member '文件名'
```

(5) 重新初始化联机重做日志组：

```
alter database clear logfile group n
```

(6) 清除崩溃的重做日志文件使其不能归档：

```
alter database clear unarchived logfile group n
```

(7) 当修改完相关日志文件后，可以使用数据字典 v$log 和 v$logfile 进行查看。

3.2.4 参数文件

通过学习 Oracle 数据库的体系结构，实例是一组 Oracle 后台进程和内存结构的集合，那么实例到底占用多大内存空间，并且在启动实例时是否要启动某些特定的后台进程，这都需要通过配置参数文件来完成。如以 sysdba 身份发出 startup 命令，Oracle 服务器就会读取初始化参数文件，并根据参数文件来配置实例。在启动实例时必须有相应的初始化参数存在。数据库参数文件主要用于保存数据库的非默认参数。

参数文件如图 3-4 所示。

1. 参数文件分类

Oracle 有两种不同类型的初始化参数文件。

(1) 静态参数文件(pfile)：该文件为正文文件。静态参数文件的文件名一般为 initSID.ora。

(2) 动态服务器参数文件(spfile)：该文件为二进制文件。动态服务器参数文件的文件名一般为 spfileSID.ora。这里的 SID 为实例名。

2. 查看参数文件

为了提高 Oracle 服务器运行性能，可能需要经常查询或修改相关参数。那么如何监视

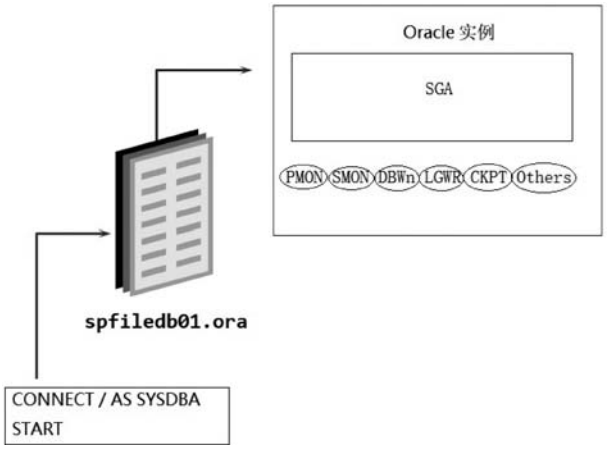

图 3-4 参数文件

初始化参数的设置呢？可通过如下方法实现。

（1）show parameter：显示当前会话中所有初始化参数的值。

```
SQL> show parameter pfile;

NAME         TYPE         VALUE
 --------  ---------- ------------------------------------
spfile         string      D:\SOFTWARE\ORACLE19C\DATABASE\SPFILENEWORCL.ORA
SQL> show parameter spfile;

NAME         TYPE         VALUE
 --------  ---------- ------------------------------------
spfile         string      D:\SOFTWARE\ORACLE19C\DATABASE\SPFILENEWORCL.ORA
```

（2）v$paramenter：包含当前会话中所有初始化参数及其值。

```
SQL> select name,value from v$parameter where name = 'spfile';

NAME       VALUE
 ------ ---------------------------------- ---------
spfile     D:\SOFTWARE\ORACLE19C\DATABASE\SPFILENEWORCL.ORA
```

3. 修改参数

修改 Oracle 参数的 SQL 语法为：

```
alter system   set parameter = value
<deferred>
<scope = both| spfile| memory>
<sid = 'sid| * '>
```

其中：parameter＝value 表示参数名和参数值；deferred 表示设置参数对当前会话不生效，对以后的会话生效；scope 表示设置参数的作用范围。其中，both 表示设置参数在当前实例中生效，并将参数修改后的值保存在 spfile 参数文件中；spfile 表示设置参数仅保存在 spfile 参数文件中，要重启才能生效；memory 表示设置参数仅作用于当前实例。

3.2.5 其他文件

(1) 跟踪文件(trace file)：是数据库中重要的诊断文件，是获取数据库信息的重要工具，对管理数据库的实例起着至关重要的作用。跟踪文件中包含数据库系统运行过程中所发生的重大事件的有关信息，可以为数据库运行故障的解决提供重要信息，每个后台进程都有相应的跟踪文件。

(2) 告警文件(alert file)：是数据库中重要的诊断文件，记录数据库在启动、关闭和运行期间后台进程的活动情况。其中跟踪文件与告警文件将在后续章节详细展开讨论。

(3) 备份文件(backup file)：是历史联机重做日志文件的集合，是联机重做日志文件被覆盖之前备份的副本。

(4) 口令文件(password file)：存放用户口令的加密文件。

3.3 Oracle 启动与关闭

各文件之间的关系如图 3-5 所示。从图中可以发现，文件之间与数据库的启动息息相关。

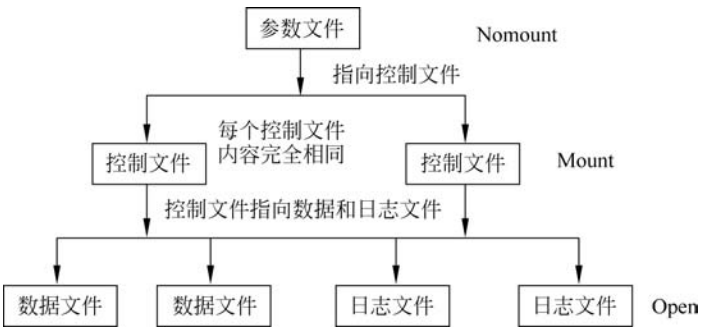

图 3-5 各文件之间的关系

由图 3-5 所示的各文件之间的关系可引入 Oracle 的物理结构，建立层次的概念，同时引入数据库启动的 3 个阶段。在讲解这部分内容的时候，一定要结合构成层次结构的 3 个层次的文件和启动过程的关系展开，从而将这些基本的概念统合到一起。

Oracle 的启动分为 3 个阶段：启动实例(Nomount)、装载数据库(Mount)和打开数据库(Open)。

Oracle 的正常关闭也分为 3 个阶段：关闭数据库(Closed)、卸载数据库(Dismounted)和终止进程(Instance Shutdown)。

下面具体了解一下 Oracle 的启动和关闭。

3.3.1 Oracle 的启动过程

Oracle 数据库的启动分为 3 个阶段:启动实例、装载数据库和打开数据库。这里的每个阶段都会读取和校验不同的数据库文件,并按照时间的先后顺序将相关信息写入到告警日志中,所以在启动等过程中如果出现问题,就可以观察和研究告警日志。这也是学习 Oracle 数据库的一个重要途径。

1. 启动实例(Nomount)

Oracle 会读取一个参数文件(PFILE 或者 SPFILE 文件),Oracle 根据参数文件中的参数,分配一系列后台进程和服务进程,并且在内存中创建 SGA 区等内存结构。内存和进程就组成了所谓的实例。每一个进程都拥有一个自己的名字(SID)。在实例启动完毕,数据库还没有跟实例关联,还处于 Nomount 状态,表明数据库还不可以访问。

实例启动的语句为:

```
SQL> startup nomount
```

在 Nomount 模式下,只能访问那些与 SGA 区相关的数据字典视图,包括 v$parameter、v$sga、v$process 和 v$session 等,这些视图中的信息都是从 SGA 区中获取的,与数据库无关。实例启动可执行重建控制文件、重建数据库,读取 init.ora 文件,启动 instance,即启动 SGA 和后台进程。这种启动只需要 init.ora 文件。

实例启动过程如图 3-6 所示。

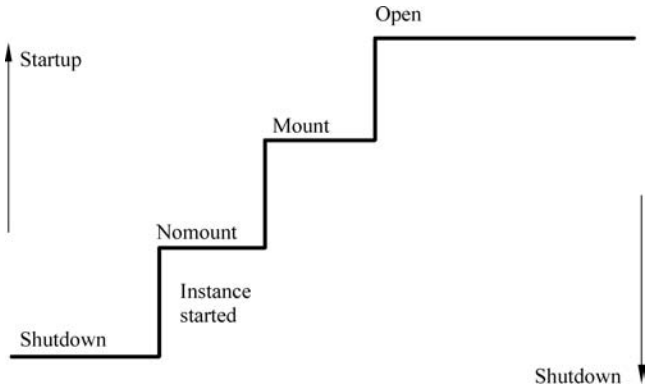

图 3-6 启动实例过程

2. 装载数据库(Mount)

装载阶段创始实例并且安装数据库,但没有打开数据库。Oracle 系统读取控制文件中关于数据文件和重做日志文件的内容,但是并不打开该文件。在这种打开方式下,系统会给出“数据库装载完毕”的提示。

装载数据库的语句为:

```
SQL> startup mount
```

这种启动模式将为实例加载数据库，但保持数据库为关闭状态。因为加载数据库时需要打开数据库控制文件，但数据文件和重做日志文件都无法进行读写，所以用户还无法对数据库进行操作。在 Mount 模式下，只能访问那些与控制文件相关的数据字典视图，包括 v$thread、v$controlfile、v$database、v$datafile 和 v$logfile 等，这些视图都是从控制文件中获取的。这种打开方式经常在数据库维护操作时使用，如数据库日志归档、数据库介质恢复、使数据文件联机或脱机、重新定位数据文件和重做日志文件。

装载数据库过程如图 3-7 所示。

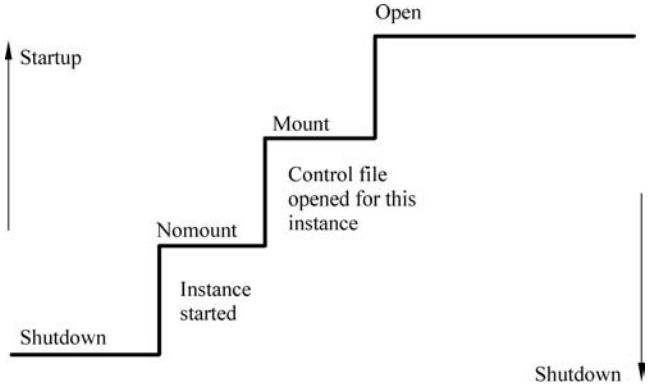

图 3-7　装载数据库过程

3. 打开数据库（Open）

在打开数据库时，实例将打开所有处于联机状态的数据文件和重做日志文件。若控制文件中的任何一个数据文件或重做日志文件无法正常打开，数据库都将会返回错误信息。这时需要进行数据库恢复。

打开数据库的语句为：

```
SQL> startup [open]
```

这个命令等同于以下 3 个命令。

```
SQL> startup nomount;
SQL> alter database mount;
SQL> alter database open;
```

只有设置为打开状态，数据库才处于正常状态，这时普通用户才能够访问数据库。在很多情况下，启动数据库时并不是直接完成上述 3 个步骤，而是逐步完成的，然后执行必要的管理操作，最后才使数据库进入正常运行状态。所以才有了各种不同的启动模式用于不同的数据库维护操作。

打开数据库过程如图 3-8 所示。

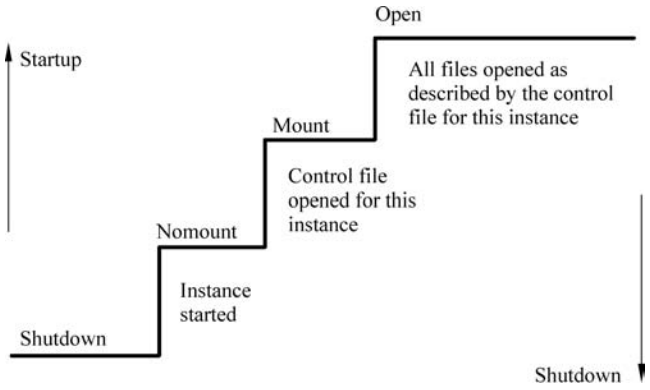

图 3-8 打开数据库过程

3.3.2 数据库的启动

因为在 Oracle 数据库启动过程中,不同的阶段可以对数据库进行不同的维护操作,对应不同的需求,所以需要不同的模式启动数据库。

1. 启动命令

上面已经简单说明了实例启动的 3 个阶段的通用启动命令,下面对启动命令语法进行详细讲解。

```
startup [force][restrict][pfile = filename]
[open [recover] [dbname]
|mount|nomount]
```

其中:

open:启动实例,装载并打开数据库,为默认选项;

mount:启动实例并装载数据库,但不打开数据库;

nomount: 启动实例,但不能装载数据库;

pfile=: 指定用于启动实例的非默认初始化参数文件名;

force: 强制中止实例,并重新启动数据库;

restrict: 启动后只允许具有 restricted session 权限的用户访问数据库;

recover: 在数据库启动时进行介质恢复。

2. 只读状态打开数据库

在正常启动状态,默认数据库进入读/写状态,在必要时可以将数据库设置为只读状态。在只读状态下,用户只能查询数据库,但不能以任何方式对数据库对象进行修改。

使用如下命令,可以使数据库进入只读状态或进入读/写状态:

```
语法 1:startup open [read write|read only]
语法 2:alter database open [read write|read only]
```

3. 用限制模式

限制模式只允许具有 restricted session 权限的用户正常使用数据库; 而其他用户被限

制使用。

使用如下命令，可以使数据库进入限制模式：

```
语法 1:alter system [{enable|disable}restricted session]
语法 2:startup restrict
```

其中：

enable：启动限制模式；

disable：取消限制模式。

提示：

当数据库切换到 restricted session 状态时，先前登录的不具有 restricted session 权限的用户仍然可以正常工作。

3.3.3 数据库的关闭过程

与数据库启动一样，关闭数据库实例也需要分 3 步：关闭数据库、卸载数据库和终止进程。

1. 关闭数据库

在该阶段，Oracle 将重做日志高速缓冲区中的内容写入重做日志文件，并且将数据库高速缓冲中被改动过的数据写入数据文件，然后再关闭所有的数据文件和重做日志文件，这时数据库的控制文件仍然处于打开状态。但是由于数据库处于关闭状态，所以用户无法访问数据库。

2. 卸载数据库

在关闭数据库后，例程才能被卸载，控制文件在这个时候被关闭，但是例程仍然存在。

3. 终止进程

进程终止，分配给例程的内存 SGA 区被回收。

3.3.4 数据库的关闭

当 DBA 要执行完数据库备份、修改初始化参数以及其他系统维护操作时，需要停止 Oracle 服务器。

数据库关闭分为两类共 4 种方式：Normal（正常关闭）、Immediate（立即关闭）、Transaction（直接关闭）、Shutdown Abort（终止关闭）。

其中，前 3 种关闭方式属于一致性数据库关闭，特点是无须进行数据库恢复；而 Shutdown Abort 关闭方式属于非一致性数据库关闭。使用 Abort 关闭、数据库发生实例故障（如断电）、或使用 Startup Force 强制重新启动数据库，都需要进行实例恢复。

一致性数据库关闭与非一致性数据库关闭如图 3-9 和图 3-10 所示。

1. Normal（正常关闭）

当使用正常方式关闭数据时，Oracle 执行如下操作。

（1）阻止任何用户建立新的连接。

（2）等待当前所有正在连接的用户主动断开连接（此方式下 Oracle 不会立即断掉当前

图 3-9 一致性数据库关闭示意

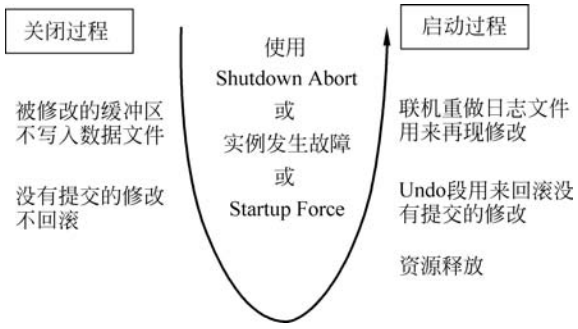

图 3-10 非一致性数据库关闭示意

用户的连接，这些用户仍然可操作相关的操作）。

（3）一旦所有的用户都断开连接，就立即关闭、卸载数据库，并终止实例。

2. Immediate（立即关闭）

当采用立即关闭数据时，Oracle 执行如下操作。

（1）阻止任何用户建立新的连接，同时阻止当前连接的用户开始任何新的事务。

（2）Oracle 不等待在线用户主动断开连接，强制终止用户的当前事务，将任何未提交的事务回退（如果存在太多未提交的事务，此方式将会耗费很长时间终止和回退事务）。

3. Transaction（直接关闭、卸载数据库，并终止实例）

这种方式介于正常关闭方式跟立即关闭方式之间，响应时间会比较快，处理也将比较得当。执行过程如下所述。

（1）阻止任何用户建立新的连接，同时阻止当前连接的用户开始任何新的事务。

（2）等待所有未提交的活动事务提交完毕，然后立即断开用户的连接。

（3）直接关闭、卸载数据库，并终止实例。

4. Shutdown Abort（终止关闭）

这是比较粗暴的一种关闭方式，当前面 3 种方式都无法关闭时，可以尝试使用终止方式来关闭数据库。但是以这种方式关闭数据库将会丢失一部分数据信息，当重新启动实例并打开数据库时，后台进程 SMON 会执行实例恢复操作。一般情况下，应当尽量避免使用这

种方式来关闭数据库。执行过程如下所述。

(1) 阻止任何用户建立新的连接,同时阻止当前连接的用户开始任何新的事务。

(2) 立即终止当前正在执行的 SQL 语句。

(3) 任何未提交的事务均不被退回。

(4) 直接断开所有用户的连接,关闭、卸载数据库并终止实例。

综上所述,可利用表 3-1 所示的来总结以下 4 种关闭模式的区别。

表 3-1 4 种关闭模式的区别

关闭模式	Shutdown Abort	Immediate	Transaction	Normal
允许新的连接	×	×	×	×
等待当前会话结束	×	×	×	√
等待当前事务结束	×	×	√	√
强制检查点并关闭文件	×	√	√	√

3.4 Oracle 逻辑存储

逻辑存储结构是从逻辑的角度来分析数据库的构成,是数据库创建后利用逻辑概念来描述 Oracle 数据库内部数据的组织和管理形式。在操作系统中,没有数据库逻辑存储结构信息,只有物理存储结构信息。数据库的逻辑存储结构信息存储在数据库的数据字典中,可以通过数据字典进行查询。

Oracle 数据库的逻辑结构是一种层次结构,主要由表空间、段、区和块等概念组成。逻辑结构是面向用户的,用户使用 Oracle 开发应用程序使用的就是逻辑结构。

Oracle 数据库逻辑存储结构之间的关系如图 3-11 所示。

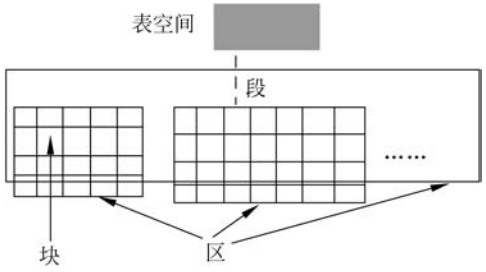

图 3-11 Oracle 数据库逻辑存储结构之间的关系

(1) 块(Block):数据库中最小的 I/O 单元。

(2) 区(Extent):由若干连续的数据块组成,是数据库中最小的存储分配单元。

(3) 段(Segment):由若干区组成,存储相同类型数据。

(4) 表空间(Tablespace):由若干段组成,是最大的存储逻辑单元。所有表空间构成数据库。

3.4.1 表空间

1. 表空间的概念

表空间(Tablespaces)是数据库的逻辑划分,一个表空间只能属于一个数据库。所有的

数据库对象都存放在指定的表空间中。由于主要存放的对象是表，所以称作表空间。

Oracle 数据库的每个表空间包含一个或者多个“.dbf”的操作系统文件，称为数据文件。数据文件的大小决定了表空间的大小。一个数据文件只能从属于一个表空间。

表空间是存储模式对象的容器，一个数据对象只能存储在一个表空间中（分区表和分区索引除外），但可以存储在该表空间所对应的一个或者多个数据文件中。若表空间只有一个数据文件，则该表空间中所有对象都保存在该文件中；若表空间对应多个数据文件，则表空间中的对象可以分布于不同的数据文件中。

表空间与数据文件、数据对象的关系如图 3-12 所示。

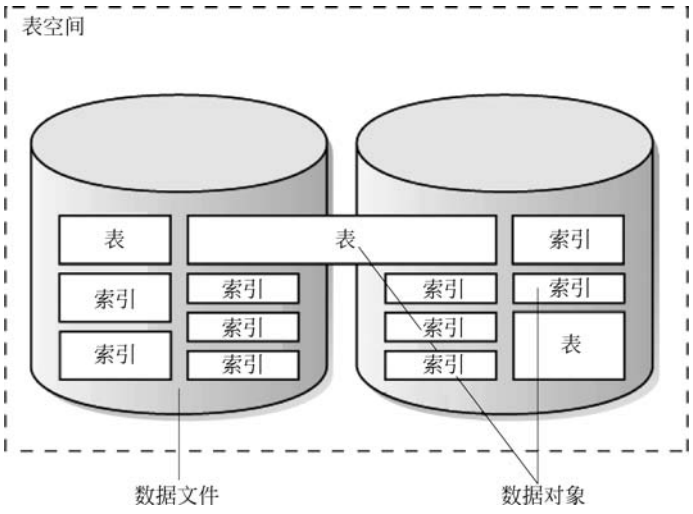

图 3-12 表空间与数据文件、数据对象的关系

表空间从逻辑上是多个段的结合，在物理上是多个数据文件的集合，相当于在段和数据文件的对应中加入了一个中间层来解决这种多对多的关系。

表空间、段、区、块之间的逻辑关系如图 3-13 所示。

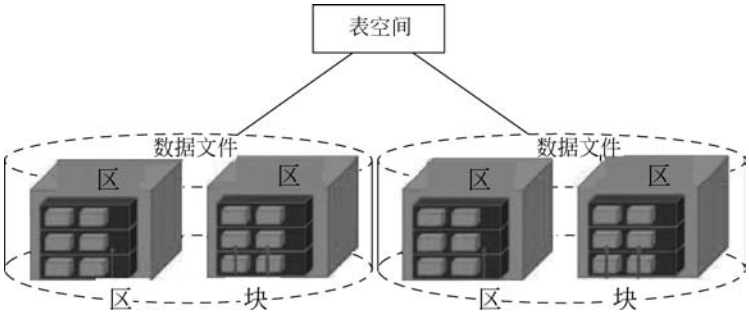

图 3-13 表空间、段、区、块之间的逻辑关系

2. 表空间的分类

数据库在安装完毕，会创建 5 个基本的表空间。关于 Oracle 表空间的分类如表 3-2 所示。

表 3-2 Oracle 表空间的分类

Tablespace 类型	用途说明
SYSTEM	存储系统数据字典的相关表、视图
SYSAUX	存放附加的数据库组件，如企业管理器资料库
TEMP	存放系统处理排序时所用的过渡型数据
UNDOTBS	存放系统执行交易回滚所需的建议前数据
USERS	用于存放用户私有信息

表空间可分为系统表空间和非系统表空间。系统表空间(包括 SYSTEM 和 SYSAUX 表空间)与数据库一起建立，为强制性表空间，必须为联机状态。而非系统(Non-SYSTEM)表空间存储一些单独的段，方便磁盘空间管理，控制分配给用户磁盘空间的数量。

3. 与表空间的相关视图

(1) v$tablespace：从控制文件中获取的表空间名称和编号信息。
(2) dba_tablespaces：数据库中所有表空间的信息。
(3) dba_tablespace_groups：表空间组及其包含的表空间信息。
(4) dba_segments：所有表空间中段的信息。
(5) dba_extents：所有表空间中区的信息。
(6) dba_free_space：所有表空间中空闲区的信息。
(7) v$datafile：所有数据文件信息，包括所属表空间的名称和编号。
(8) v$tempfile：所有临时文件信息，包括所属表空间的名称和编号。
(9) dba_data_files：数据文件及其所属表空间信息。
(10) dba_temp_files：临时文件及其所属表空间信息。
(11) dba_users：所有用户的默认表空间和临时表空间信息。
(12) dba_ts_quotas：所有用户的表空间配额信息。
(13) v$sort_segment：数据库实例的每个排序段信息。
(14) v$sort_user：用户使用临时排序段信息。

4. 表空间的操作

下面将重点介绍表空间信息的查询的命令。关于如何管理表空间，会在后续的空间管理章节展开具体讲解。

(1) 查看表空间，其代码如下：

```
SQL> select * from v$tablespace;

      TS# NAME      INC BIG FLA ENC    CON_ID
--------- -------- --- -- --- --- --------
      1    SYSAUX    YES NO  YES        0
      0    SYSTEM    YES NO  YES        0
      2    UNDOTBS1  YES NO  YES        0
      4    USERS     YES NO  YES        0
      3    TEMP      NO  NO  YES        0
```

（2）查看详细数据文件，其代码如下：

```
SQL> col tablespace_name format a10
SQL> select file_name,tablespace_name from dba_data_files;

FILE_NAME                                          TABLESPACE
-------------------------------------------------- ----------
D:\ORACLE19C\ORADATA\NEWORCL\SYSTEM01.DBF           SYSTEM
D:\ORACLE19C\ORADATA\NEWORCL\SYSAUX01.DBF           SYSAUX
D:\ORACLE19C\ORADATA\NEWORCL\USERS01.DBF            USERS
D:\ORACLE19C\ORADATA\NEWORCL\UNDOTBS01.DBF          UNDOTBS1
```

（3）查询表空间基本信息，其代码如下：

```
SQL> select tablespace_name,exetnt_management,allocation_type,contents
  2  from dba_tablespaces;

TABLESPACE  EXTENT_MAN  ALLOCATION  CONTENTS
--------- ---------  ---------- --------------------------
SYSTEM      LOCAL       SYSTEM      PERMANENT
SYSAUX      LOCAL       SYSTEM      PERMANENT
UNDOTBS1    LOCAL       SYSTEM      UNDO
TEMP        LOCAL       UNIFORM     TEMPORARY
USERS       LOCAL       SYSTEM      PERMANENT
```

（4）查询表空间数据文件信息，其代码如下：

```
SQL> select file_name,blocks,tablespace_name
  2  from dba_data_files;

FILE_NAME                                          BLOCKS   TABLESPACE
-------------------------------------------------- -------  ---------
D:\ORACLE19C\ORADATA\NEWORCL\SYSTEM01.DBF           116480   SYSTEM
D:\ORACLE19C\ORADATA\NEWORCL\SYSAUX01.DBF           92160    SYSAUX
D:\ORACLE19C\ORADATA\NEWORCL\USERS01.DBF            640      USERS
D:\ORACLE19C\ORADATA\NEWORCL\UNDOTBS01.DBF          8320     UNDOTBS1
```

（5）查询表空间空闲空间大小，其代码如下：

```
SQL> select tablespace_name,sum(bytes)free_spaces
  2  from dba_free_space
  3  group by tablespace_name;
```

```
TABLESPACE                     FREE_SPACES
------------------------------ -----------------------------------
SYSTEM                         8716288
SYSAUX                         71892992
UNDOTBS                        131850496
USERS                          2424832
```

(6) 统计表空间空闲空间信息,其代码如下:

```
SQL> select tablespace_name "tablespace", file_id, count( * )"fieces", max(blocks) "maximum",
  2  min(blocks) "minimun", avg(blocks) "avgerage", sum(blocks) "total"
  3  from dba_free_space
  4  group by tablespace_name, file_id;

TABLESPACE      FILE_ID    FIECES    MAXIMUM    MINIMUN  AVGERAGE      TOTAL
------------ --------- -------- -------- ------- ---------- -------------
SYSAUX          3          117       4736       8        75.008547     8776
UNDOTBS         14         8         3328       16       486           3888
SYSTEM          1          2         1024       40       532           1064
USERS           7          1         296        296      296           296
```

3.4.2 段

段(Segment)是由一个或多个连续或不连续的区组成的逻辑存储单元,用于存储特定的、具有独立存储结构的数据库对象。一般情况下,一个数据库对象拥有一个段。

根据存储对象类型的不同,可将段分为表段、索引段、临时段和还原段 4 类。

(1) 表段又称数据段,用来存储表或簇的数据,可以细分为普通表段(Table)、分区表段(Table Partition)、簇段(Cluster)、索引化表段(Index-Organized Table)。

(2) 索引段(Index Segment)用来存放索引数据,包括 rowid 和索引值。

(3) 临时段(Temporary Segment)是在进行查询、排序等操作时,如果内存空间不足,用于保存 SQL 语句在解释和执行过程中产生的临时数据。在会话结束时,为该操作分配的临时段将被释放。

(4) 还原段(Undo Segment)用于保存数据库的回退信息,包含当前未提交事务所修改的数据的原始版本。利用回退段中保存的回退信息,可以实现事务回滚、数据库恢复、数据的读一致性和闪回查询。

3.4.3 区

区(Extent)是数据库存储空间分配的一个逻辑单位,Oracle 数据库在分配空间时,并不是以块为单位进行的,而是多个连续的块一次性地分配给数据库对象。这些连续的块就是区。

当创建一个数据库对象时，Oracle 数据库为这些对象创建一个段，并分配初始区。当段中的初始区的存储空间使用完毕后，Oracle 数据库会为段自动分配新的区，且每个区的大小不要求相同。

3.4.4 块

1. 块的定义

块(BLock)是 Oracle 数据库管理数据文件中存储空间的单位，为数据库使用的 I/O 的最小单位，其大小可不同于操作系统的标准 I/O 块大小。

在 Oracle 19c 数据库中，块分为标准块和非标准块两种。其中，标准块由数据库初始化参数 db_block_size 设置，其大小不可更改。Oracle 数据库的默认数据缓冲区就是标准块构成的。块尺寸是处理 Oracle 数据库更新、选择或者插入数据事务的最小单位。通过下列查询，可观察默认标准块的大小为 8192B，即 8KB。

```
SQL> show parameter db_block_size

NAME                                 TYPE         VALUE
------------------------------ ------ -----------
db_block_size                         integer       8192
SQL> select value from v$parameter where name = 'db_block_size';
VALUE
------------------------------------
8192
```

2. 块的结构

Oracle 中块的结构如图 3-14 所示。

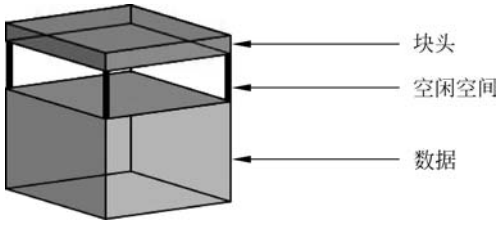

图 3-14 块的结构示意

块由块头、数据区和空闲区 3 个部分组成。其中，上面是块的头部分，下面是块的数据部分，而中间为空闲区。头部从上往下增长，而数据部分则从下往上增长，当两部分接触时块就满了。

在块中各个区域的存储信息如图 3-15 所示。

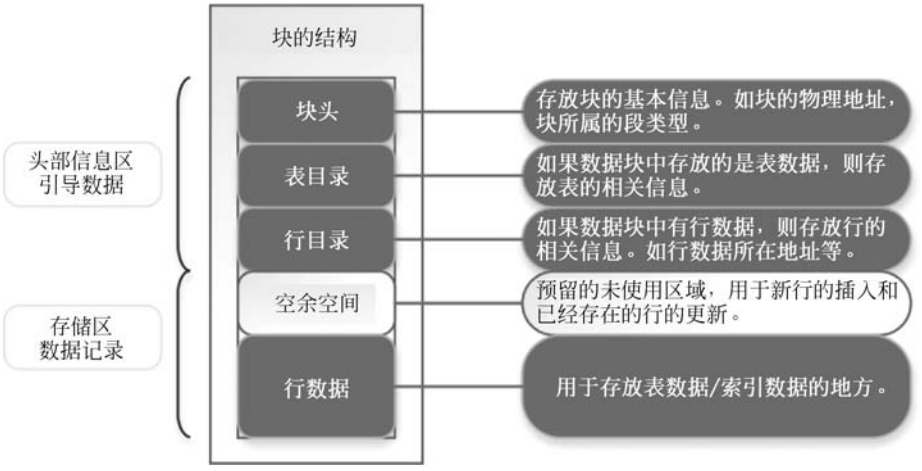

图 3-15　在块中各个区域的存储信息

3.5　项目案例


视频讲解

3.5.1　内存结构实例

通过以下案例来验证 SGA 中 database buffers 和 shared pool 的作用，具体的操作步骤如下所示。

步骤 1　设定 SQL Plus 显示时间(注意，区别于 set time on)。

```
SQL > set timing on;
```

步骤 2　第一次执行。

```
SQL > select count( * ) from dba_objects;
```

步骤 3　第二次执行。

```
SQL > select count( * ) from dba_objects;
```

步骤 4　清空共享缓存。

```
SQL > alter system flush shared_pool
```

步骤 5　清空数据库块缓存。

```
SQL > alter system flush buffer_cache
```

3.5.2　日志文件操作实例

该实验以日志文件为例，介绍日志文件的基本操作，包括创建/删除日志文件组、添加/

删除日志文件、查看日志文件/文件组、强制日志切换等内容。

步骤 1 删除重做日志文件组 3。

```
SQL> alter database drop logfile group 3;

数据库已更改.
```

步骤 2 向文件组 2 中添加一个日志文件。

```
SQL> alter database add logfile member
'D:\ORACLE19C\ORADATA\NEWORCL\REDO02 - 2.LOG' to group 2;

数据库已更改.
```

步骤 3 添加重做日志文件组 3,该文件组由 3 个日志文件组成。

```
SQL> alter database add logfile group 3
('D:\ORACLE19C\ORADATA\NEWORCL\REDO03 - 1.LOG','D:\ORACLE19C\ORADATA\NEWORCL\REDO03 - 2.LOG','D:\
ORACLE19C\ORADATA\NEWORCL\REDO03 - 3.LOG') size 15MB;

数据库已更改.
```

步骤 4 查看当前日志情况。

```
SQL> select group#,sequence#,members,archived,status,first_time from v$log;

    GROUP#  SEQUENCE#    MEMBERS  ARC  STATUS           FIRST_TIME
---------- ---------- -------- --- ------------  ---------------
         1          4          1    NO   CURRENT          15 - 2 月  - 20
         2          2          2    YES  INACTIVE         12 - 2 月  - 20
         3          0          3    YES  UNUSED
```

步骤 5 进行重做日志文件的切换。

```
SQL> alter system switch logfile;

系统已更改.
```

步骤 6 观察日志文件组状态的变化。

```
SQL> select group#,sequence#,members,archived,status,first_time from v$log;

    GROUP#  SEQUENCE#    MEMBERS  ARC  STATUS           FIRST_TIME
---------- ---------- -------- ---- ------------ ----------------
         1          4          1    YES  ACTIVE           15 - 2 月  - 20
         2          2          2    YES  INACTIVE         12 - 2 月  - 20
         3          5          3    NO   CURRENT          15 - 2 月  - 20
```

步骤 7　观察日志文件状态的变化。

```
SQL> select * from v$logfile;

GROUP#  STATUS  TYPEMEMBER                                          IS_    CON_ID
------- ------- --------------- --- ---------- ------------------ --------
2       INVALID  ONLINE D:\ORACLE19C\ORADATA\NEWORCL\REDO02 - 2.LOG  NO       0
2                ONLINE D:\ORACLE19C\ORADATA\NEWORCL\REDO02.LOG      NO       0
1                ONLINE D:\ORACLE19C\ORADATA\NEWORCL\REDO01.LOG      NO       0
3                ONLINE D:\ORACLE19C\ORADATA\NEWORCL\REDO03 - 1.LOG  NO       0
3                ONLINE D:\ORACLE19C\ORADATA\NEWORCL\REDO03 - 2.LOG  NO       0
3                ONLINE D:\ORACLE19C\ORADATA\NEWORCL\REDO03 - 3.LOG  NO       0

已选择 6 行.
```


视频讲解

3.5.3　参数文件操作实例

下面以 log_buffer 参数为例，通过实验来介绍修改该参数的具体操作步骤。

步骤 1　查看该参数的值。

```
SQL> show parameter log_buffer;

NAME                                 TYPE         VALUE
------------------------------------ ---------- ------------------------
log_buffer                           big integer  10MB
SQL> select name,value,display_value from v$parameter where name = 'log_buffer';
NAME         VALUE                                                DISPLAY_VALUE
------------ ---------------------------------------------------- -------------
log_buffer   10485760                                              10MB
```

步骤 2　修改该参数的值。

```
SQL> alter system set log_buffer = 7232KB scope = spfile;

系统已更改.
```

步骤 3　验证修改后的值。

```
SQL> show parameter log_buffer;

NAME                                 TYPE         VALUE
------------------------------------ ---------- ------------------
log_buffer                           big integer  10MB
```

该参数的值未修改的原因是作用范围为 spfile，必须重启系统后才能看到调整后的结果。

3.5.4 数据库打开与关闭实例

通过下面实验来验证数据库的打开流程以及多种关闭方式。

1. Shutdown Immediate 关闭模式测试步骤


视频讲解

步骤 1 在第一个用户进程命令窗口中启动数据库到 Open 阶段，以用户 hr 登录，在该用户中创建表 t1，并插入数据未提交。

```
SQL> conn hr
输入口令:
已连接.
SQL> create table t1(a int);
表已创建.
SQL> insert into t1  values(1);
已创建 1 行.
```

步骤 2 打开第二个用户进程命令窗口，以用户 sys 登录并关闭数据库 Shutdown Immediate。

```
C:\WINDOWS\system32>set oracle_sid = neworcl
C:\WINDOWS\system32>sqlplus
SQL PLUS: Release 19.0.0.0.0 - Production on 星期六 2 月 15 19:45:33 2020
Version 19.3.0.0.0
Copyright (c) 1982, 2019, Oracle.  All rights reserved.
请输入用户名:  sys as sysdba
输入口令:
连接到:
Oracle Database 19c Enterprise Edition Release 19.0.0.0.0 - Production
Version 19.3.0.0.0

SQL> shutdown immediate
数据库已经关闭.
已经卸载数据库.
ORACLE 例程已经关闭.
```

步骤 3 重新启动数据库，查询表 t1 数据的情况，发现已经丢失。

```
SQL> startup
ORACLE 例程已经启动.

Total System Global Area 2550136752 bytes
```

```
Fixed Size                  9031600 bytes
Variable Size             570425344 bytes
Database Buffers         1962934272 bytes
Redo Buffers               7745536 bytes
数据库装载完毕.
数据库已经打开.
SQL> select * from hr.t1;
未选定行
```

2. Shutdown Abort 关闭模式测试步骤

步骤 1　打开第一个 Command 窗口，以用户 hr 登录，向表 t1 中插入数据并提交。

```
SQL> conn hr
输入口令:
已连接.
SQL> insert into t1  values(10);
已创建 1 行.
SQL> commit;
提交完成.
```

步骤 2　打开第二个 Command 窗口，以用户 hr 登录，插入数据未提交。

```
SQL> conn hr
输入口令:
已连接.
SQL> insert into t1  values(20);
已创建 1 行.
```

步骤 3　打开第三个 Command 窗口，以用户 sys 登录，运行 Shutdown Abort 命令关闭数据库，此时会跳过前两个阶段并直接关闭实例。

```
SQL> shutdown abort;
Oracle 例程已经关闭.
```

步骤 4　重启第四个 Command 窗口，以用户 hr 登录，查看表 t1 数据情况。查询结果显示，最初插入的数据 10 存在，后续插入的数据 20 不在。这说明数据 10 即使没来得及从缓存中存入数据文件，commit 也已经将修改记录在 redo log 中，所以在重启过程中进行实例恢复，不会丢失数据。

```
SQL> startup
Oracle 例程已经启动.

Total System Global Area 2550136752 bytes
```

```
Fixed Size                  9031600 bytes
Variable Size             570425344 bytes
Database Buffers         1962934272 bytes
Redo Buffers                7745536 bytes
数据库装载完毕.
数据库已经打开.
SQL> conn hr
输入口令:
已连接.
SQL> select * from hr.t1;
A
--------
10
```

其中,在重启的过程中,系统会进行实例恢复,恢复的过程全部记录在告警文件中。在后面章节中会具体介绍告警文件和跟踪文件。