管理实例

1. 实例概述

实例是什么?

- Oracle实例是访问Oracle数据库的方法。
- 一个实例只能打开一个数据库。
- 实例是由内存SGA和后台进程组成的。
- 一个数据库就是在磁盘上是一堆物理文件组成的。实例连接上数据库后，可以帮助用户去管理数据库。

在单实例中，实例名与数据库名一般是相同的，但也可以不同，因为实例与数据库是相互独立的。

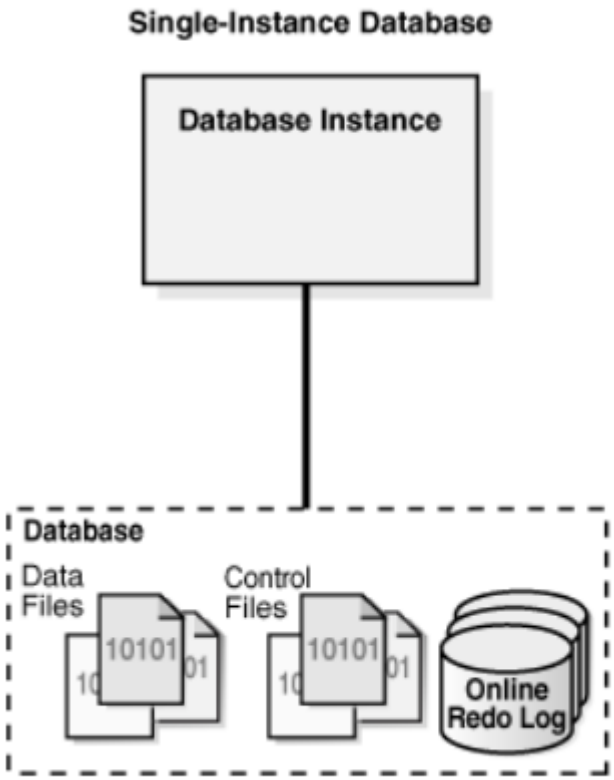
SGA主要的作用

1）将数据库文件中的内容缓存到内存 2）维护Oracle内部的数据结构信息 3）缓存重做日志条目 4）存储SQL执行计划

2. 单实例和多实例

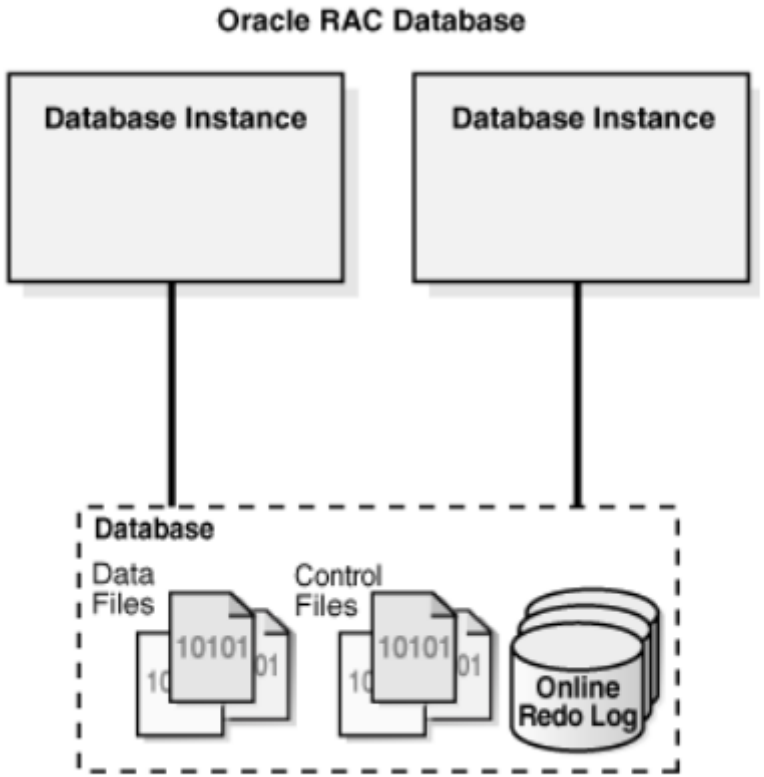
单实例

- 一个实例对应一个数据库



多实例

- 即RAC(Real Application Clusters), 多个实例对应一个数据库



3. 启动实例的认证模式

数据库最高账号sys的密码文件认证模式

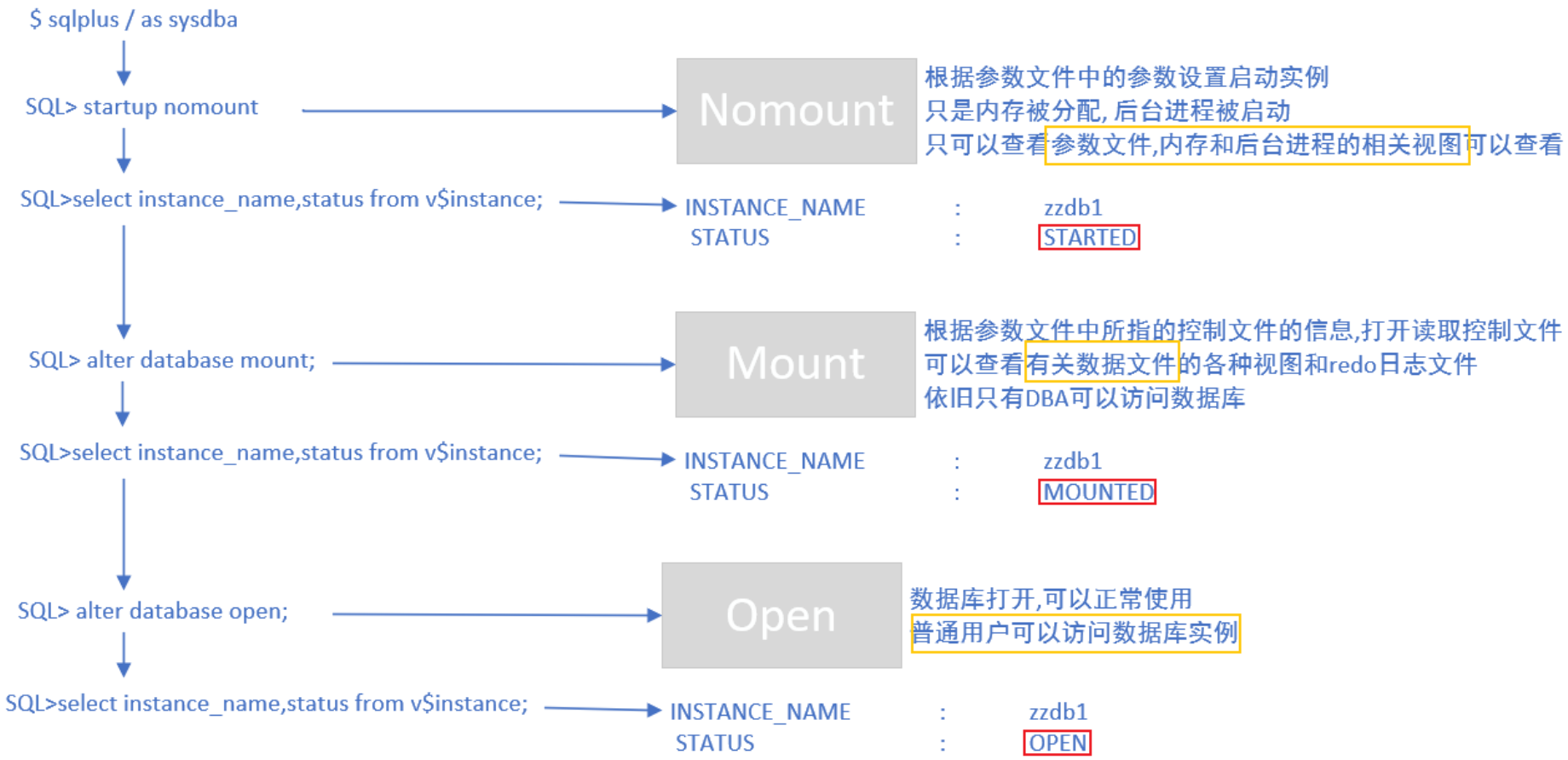
1. sysdba或sysoper本地操作系统认证登录
2. sysdba或sysoper远程客户端通过密码文件认证登录

启动实例需要的权限

- 操作系统管理员权限, Linux或UNIX下, 具有属于DBA组或oinstall组的OS用户通过 sqlplus / as sysdba 连接后, 输入startup启动
- 在Oracle系统中, 授予SYSDBA或SYSOPER权限的oracle用户, 远程以密码方式连接 sqlplus sys/sysz_password@service_name as sysdba
- 能远程以sysdba连接的条件
  - 密码文件存在
    - 密码文件存放的是sys账号的密码
    - 建立密码文件要重新启动数据库, 因为内存中保留有原来的密码
    - 初始化参数remote_login_passwordfile=none则数据库设置为禁止使用密码文件
    - 创建密码文件

      ```
      orapwd file=$ORACLE_HOME/dbs/orapwSID password=123456 force=y
      ```
  - 参数remote_login_passwordfile设置为EXCLUSIVE

4. 实例启动的三个阶段



5. startup

startup命令语法

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

数据库只读模式

默认打开数据库是读写模式。只读模式，数据库只能读，无法写入、更新等操作。 • 数据文件可以offline或online，但永久表空间不能offline。 • Offline的数据文件可以恢复。 • 控制文件可以更新修改，其记录的是数据库的状态。 • 临时表空间可以被创建，不涉及数据库数据。 • Alter日志、trace文件、audit文件可以正常进行。

数据库可以以只读方式打开

```
SQL> startup mount;
SQL> alter database open read only;
```

6. shutdown

关闭模式	Abort	Immediate	Transactional	Normal
允许建立新连接	否	否	否	否
等待当前会话结束	否	否	否	是
等待当前事务结束	否	否	是	是
强制执行检查点和关闭文件	否	是	是	是

默认值

NORMAL

正常关闭数据库，在 Normal 关闭数据库时，除了不允许建立新的连接外，允许当前正在进行的会话继续进行，当前正在进行的事务，也可正常运行直到终止。 例子：（两个会话，以 NORMAL 方式关闭，必须等待另一会话退出，另一会话中仍可以执行 DML、Select、DDL等语句，另一会话退出，数据库才会关闭）。 在所有的会话正常退出后，Normal 才会关闭数据库。 在等待期间，不能有新的连接，不过仍然可以有新的事务。

TRANSACTIONAL

会等待所有的事务结束后，就关闭数据库。无论会话是否退出。 在TRANSACTIONAL完成关闭操作期间，不允许有的事务，更不允许连接。

IMMEDIATE

ORACLE会自动回退没有提交的事物，断开所有连接，关闭所有没有结束的会话，立即关闭数据库。

> 以上 3 种关闭方式，在数据库关闭前，都会**先将 buffer cache 中的脏数据刷新至磁盘文件**，这个操作在 Oracle 中被称为**完全检查点**。因为所有的脏数据都被写进磁盘，因此，**不会丢失用户所做的修改**。下一次启动时也不需要利用日志进行恢复。我们称这种方式关闭的数据库是一致的数据库，也称为干净的数据库。

ABORT

关闭数据库的话，ORACLE 会立即退出，并不刷新脏数据到磁盘。 Abort 时的操作，就好象断电和死机一样，Oracle立即中止。

我们通常可以**利用这种方式来模拟数据库出现故障后的情况**，来熟悉如死机、掉电等突发情况后的 Oracle 的恢复。 ABORT 关闭数据库后，数据库将是不一致的或着是脏的，下次启动时是需要读取日志来进行恢复的。

controlfiles

- 内容

    - 数据库名字
    - 数据库创建时间
    - 表空间名字
    - 数据文件和日志文件的路径
    - 当前redo日志序列号检查
    - 检查点信息
    - undo信息
    - redo日志归档信息
    - 备份信息
- 查询某些文件空间的总数以及已使用的数量

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

```
SQL> l
  1* select records_total,records_used from v$controlfile_record_section where type='DATABASE'
SQL> select * from v$controlfile_record_section;

TYPE                 RECORD_SIZE RECORDS_TOTAL RECORDS_USED FIRST_INDEX LAST_INDEX LAST_RECID
-------------------- ----------- ------------- ------------ ----------- ---------- ----------
DATABASE                     316             1            1           0          0          0
CKPT PROGRESS               8180             4            0           0          0          0
REDO THREAD                  256             1            1           0          0          0
REDO LOG                      72            16            3           0          0          3
DATAFILE                     520          1000            4           0          0          4
FILENAME                     524          4098           11           0          0          0
TABLESPACE                    68          1000            5           0          0          5
TEMPORARY FILENAME            56          1000            1           0          0          1
RMAN CONFIGURATION          1108            50            0           0          0          0
LOG HISTORY                   56           292           69           1         69         69
OFFLINE RANGE                200          1063            0           0          0          0

TYPE                 RECORD_SIZE RECORDS_TOTAL RECORDS_USED FIRST_INDEX LAST_INDEX LAST_RECID
-------------------- ----------- ------------- ------------ ----------- ---------- ----------
ARCHIVED LOG                 584            28            0           0          0          0
BACKUP SET                    40          1227            0           0          0          0
BACKUP PIECE                 736          1000            0           0          0          0
BACKUP DATAFILE              200          1063            0           0          0          0
BACKUP REDOLOG                76           215            0           0          0          0
DATAFILE COPY                736          1000            0           0          0          0
```

- 查询控制文件的名称位置和状态

```
SQL> select * from v$controlfile
```

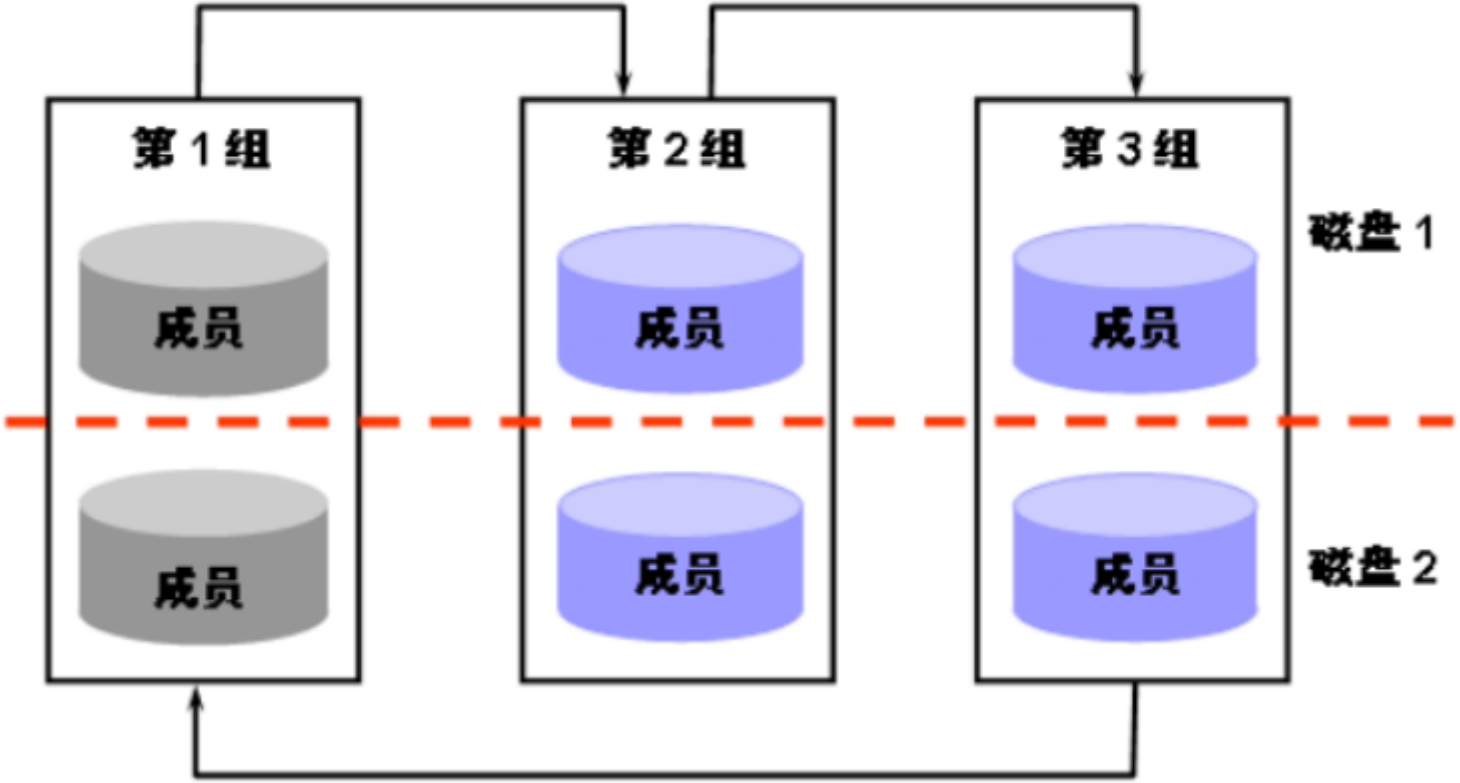
```
SQL> select * from v$controlfile;

STATUS  NAME                                               IS_ BLOCK_SIZE FILE_SIZE_BLKS
------- -------------------------------------------------- --- ---------- --------------
        /oradata/zzdb1/control01.ctl                       NO       16384           1056

SQL>
```

- 增加/删除控制文件

```
# 修改spfile中的参数
# # 增加即将参数值在原有的文件名上再加入一个文件名
# # 删除即在原有参数值上删掉要删除的文件名
SQL> alter system set control_files='/oradata/zzdb1/control01.ctl','/oradata/zzdb1/control02.ctl' scope=spfile;

# 关闭数据库
SQL> shutdown immediate

# 复制或删除对应文件
SQL> !cp /oradata/zzdb1/control01.ctl /oradata/zzdb1/control02.ctl
SQL> !mv /oradata/zzdb1/control02.ctl /oracle

 # 开启数据库
SQL> startup

# 查看是否修改成功
SQL> show parameter control_files;
```

> 问题:
>
> > 当数据库处于open状态时, 操作系统端误删除某一个控制文件, 此时正常关闭数据库会报错, 如何恢复正常?
>
> 解决:
>
> > **shutdown abort后将缺少的控制文件复制恢复(有问题)**

redo logs



基本概念

- 一组相同的联机重做日志文件副本称作online redo log group
- 组内的每个联机重做日志文件称为member
- online redo log group和member的初始化是在**数据库创建时**创建的
- 重做日志文件：online和archived两种。

联机重做日志文件 Online redo log file

- 记录了数据的变化
- 提供了恢复机制
- 以组的形式
- 至少需要两组

```
# 查看是否可用与文件名
SQL> select * from v$logfile;

    GROUP# STATUS  TYPE    MEMBER                          IS_
---------- ------- ------- ------------------------------  ---
     1       ONLINE  /oradata/zzdb1/redo01a.log              NO
     1       ONLINE  /oradata/zzdb1/redo01b.log              NO
     2       ONLINE  /oradata/zzdb1/redo02a.log              NO
     2       ONLINE  /oradata/zzdb1/redo02b.log              NO
     3       ONLINE  /oradata/zzdb1/redo03a.log              NO
     3       ONLINE  /oradata/zzdb1/redo03b.log              NO

# 查看状态等相关信息
SQL> select group#,status,sequence#,blocksize,members from v$log;

GROUP#     STATUS    SEQUENCE#   BLOCKSIZE     MEMBERS
------ ----------  ----------  ----------   --------
     1   INACTIVE            70          512           2
     2    CURRENT           71          512           2
     3   INACTIVE            69          512           2
```

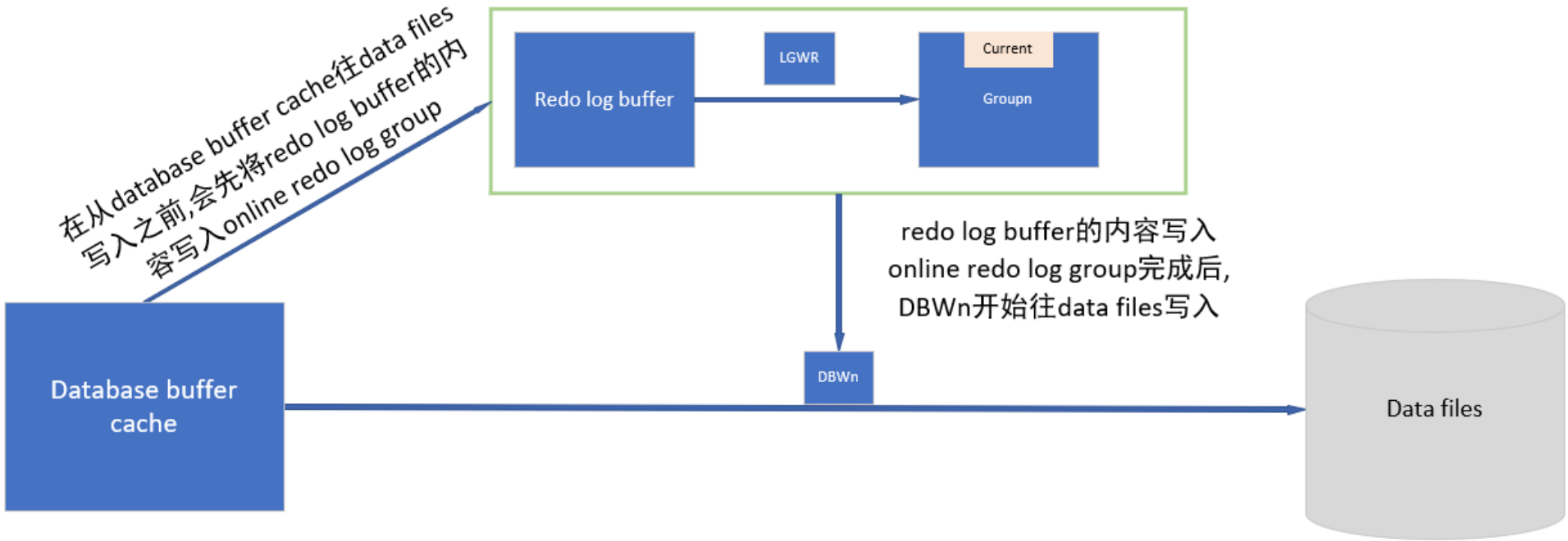
操作

```
# 增加:增加后需切换当前log才可用
# # 增加group
SQL> alter database add logfile group 3('/oradata/zzdb1/redo03a.log','/oradata/zzdb1/redo03b.log')
   / size 4m blocksize 512 reuse;
# # 增加member
SQL> alter database add logfile member '/oradata/zzdb1/redo01b.log' to group 1;
# 删除:当前组不可删除,删除后组数不可少于两组, 最后一个成员不可删除
# # 删除group
SQL> alter database drop logfile group 2;
# # 删除member
alter database drop logfile member '/oradata/zzdb1/redo01b.log';
# 清空log:当前log不可删除
SQL> alter database clear logfile group 2;
# 切换当前log:不建议轻易使用
SQL> alter system switch logfile;
```

> 问题 : 生产环境中如何增加组与成员?

日志切换

- 重做日志文件是以**循环方式**使用的。一旦某个重做日志文件被写满，LGWR就会移动到下一个日志组，这称为日志切换，同时还将执行检查点操作，将信息写入控制文件。
- Oracle服务器将对数据库所做的所有更改按顺序记录到redo log buffer中。 LGWR进程把 redo entries 从 redo log buffer 写入 online redo log group 的其中一个组，这个组叫做The current online redo log group
- 日志切换方式
  - 自动切换：日志写满oracle会写下一个组
  - 手工切换：alter system switch logfile;

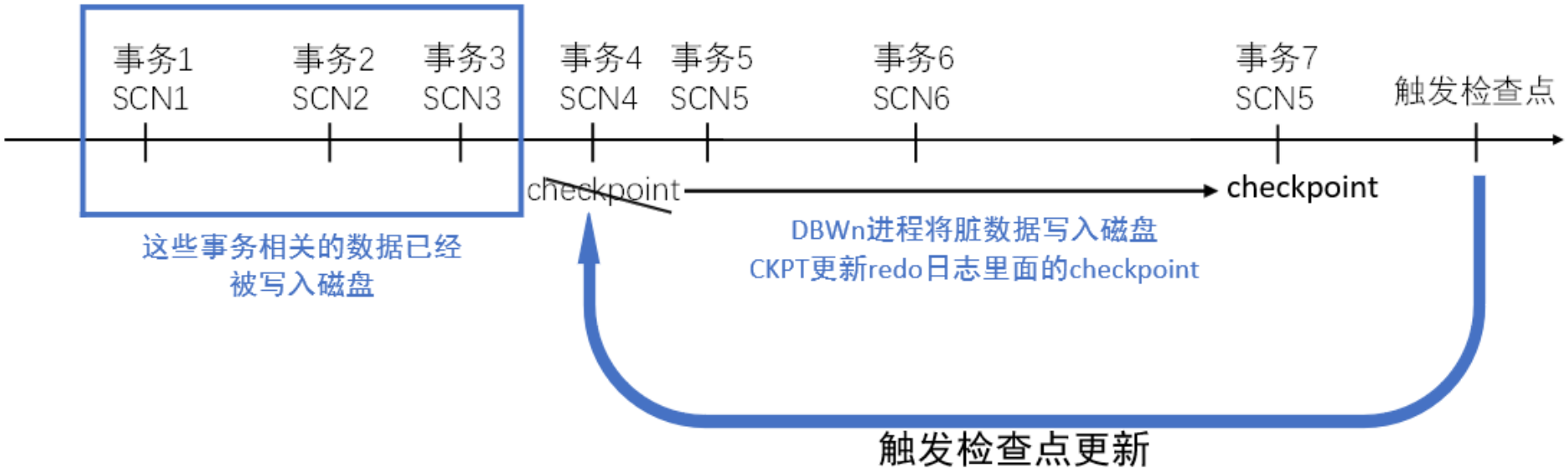


检查点

- 完全检查

  1. 一致性shutdown数据库，除了abort之外的其他三种关闭方式。
  2. alter system checkpoint; 完全检查后，所有的脏数据块都写入数据文件，改写文件的头。

- 增量检查 除了完全检查点以外的所有其它检查点都是增量检查点，增量检查是查找检查点列表，将某一个时间点做标记，该时间点前的脏块写入到数据文件，增量检查不一定马上执行，根据脏的块多少来决定，这就出现了检查点滞后的情况。



> 问题: 什么时候事务4被包含什么时候不被包含

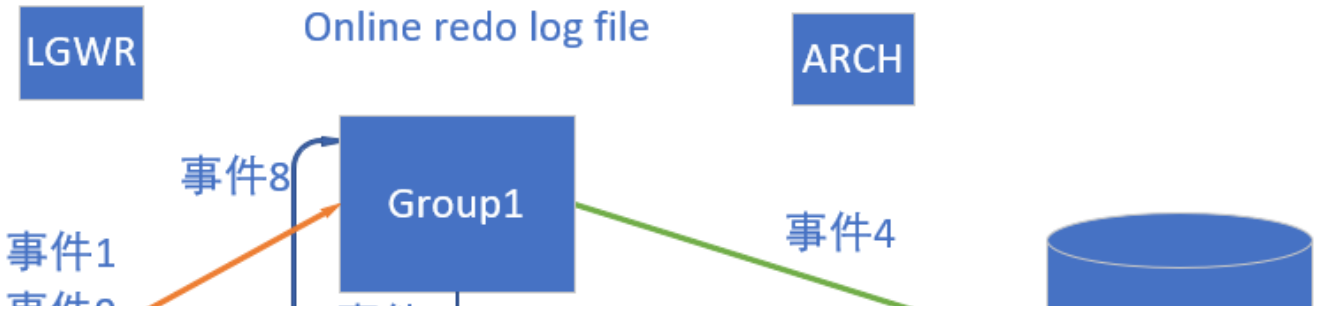
- FAST_START_MTTR_TARGET参数指定了实例异常崩溃后恢复需要的秒数。
- log_checkpoints_to_alert决定是否将检查点的信息写入alert日志。默认为假，不写日志。

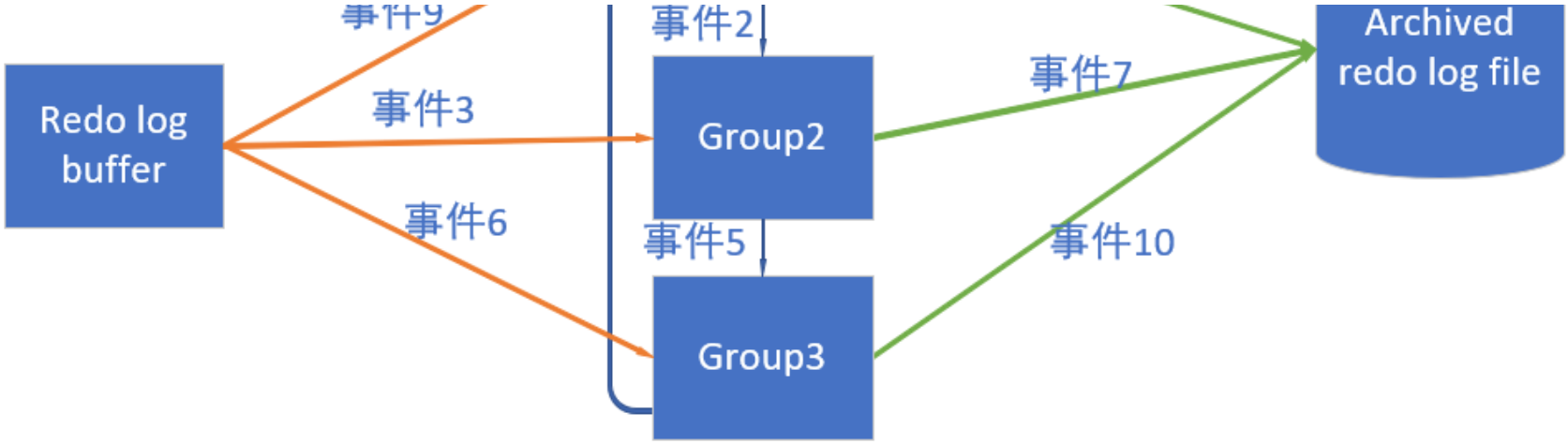
归档重做日志文件Archived redo log file

- 文件系统异常，使用**归档重做日志**以及**在线重做日志**及**备份数据文件**将**数据库恢复**到适当的时间点。
- 业务人员操作失误，例如，误删除表，**可将数据库恢复到删除表之前的某一个时间点。**

- NOARCHIVELOG模式 在NOARCHIVELOG模式下，每次联机重做日志文件已满并发生日志切换时，都要**覆盖联机重做日志文件。** 直到对重做日志文件组的检查点操作完成后，LGWR才覆盖该重做日志文件组。
- ARCHIVELOG模式 如果数据库配置为在 ARCHIVELOG 模式运行下，那么必须将**已满的**联机重做日志文件的**不活动组**归档。 因为对**数据库所做的所有更改都记录在联机重做日志文件**内，数据库管理员可以使用**物理备份**和**归档的联机重做日志文件**恢复数据库，而不会丢失任何**已提交数据。**
- 归档模式开启

```
# 开启归档
# # 开启之前创建归档的文件夹
SQL> shutdown immediate;
SQL> startup mount;
SQL> alter database archivelog;
SQL> alter system set log_archive_dest_1='LOCATION=/oradata/archive';
SQL> alter database open;
# 关闭归档
SQL> shutdown immediate;
SQL> startup mount;
SQL> alter database noarchivelog;
# SQL> alter system set log_archive_dest_1='LOCATION=/oradata/archive';
SQL> alter database open;
```





事件详情:

事件1:LGWR将redo log buffer里的redo entries写入group1

时间1:

事件2:group1被写满,日志切换到group2

事件3:LGWR开始对group2进行写入

事件4:AHCE将写满的group1进行归档

时间2:

事件5:group2被写满,日志切换到group3

事件6:LGWR开始对group3进行写入

事件7:AHCE将写满的group2进行归档

时间3:

事件8:group3被写满,日志切换到group2

事件9:LGWR开始对group1进行覆盖写入

事件10:AHCE将写满的group3进行归档

事件2,5,8发生时同时触发检查点

附加

数据库审计

数据库审计是对数据库访问行为进行监管的系统，一般采用旁路部署的方式，通过镜像或探针的方式采集所有数据库的访问流量，并基于SQL语法、语义的解析技术，记录下数据库的**所有访问和操作行为**，例如**访问数据的用户**（IP、账号、时间），**操作**（增、删、改、查）、**对象**（表、字段）等。

数据库审计系统的主要价值有两点:

1. 在发生数据库安全事件（例如数据篡改、泄露）后为事件的追责定责提供依据
2. 针对数据库操作的风险行为进行时时告警。

控制参数 : AUDIT_TRAIL 默认值为NONE

AUDIT_TRAIL = { none | os | db | db_extended | xml | xml_extended }

- none 禁用数据库审计,默认值

- os 启用数据库审计，并将数据库审计记录定向到操作系统审计记录
- db 启用数据库审计，并将数据库所有审计记录定向到数据库的SYS.AUD$表
- db_extended 启用数据库审计，并将数据库所有审计记录定向到数据库的SYS.AUD$ 表。另外，填充SYS.AUD$表的SQLBIND 列和SQLTEXT CLOB 列。
- xml 启用数据库审计，并将所有记录写到XML格式的操作系统文件中。
- xml_extended 启用数据库审计，输出审计记录的所有列，包括SqlText和SqlBind的值。

Temporary tablespace 临时表空间

临时表空间主要用途是在**数据库进行排序运算**[如创建索引、order by及group by、distinct、union/intersect/minus/、sort-merge及join、analyze命令]、**管理索引**[如创建索引、IMP进行数据导入]、**访问视图**等操作时**提供临时的运算空间**，当运算完成之后**系统会自动清理。**

当临时表空间不足时，表现为运算速度异常的慢，并且临时表空间迅速增长到最大空间（扩展的极限），并且一般不会自动清理了。

如果临时表空间没有设置为自动扩展，则临时表空间不够时事务执行将会报**ora-01652** 无法扩展临时段的错误

解决方法:

1、设置临时数据文件自动扩展

2、增大临时表空间。

脚本说明

catalog.sql

catalog.sql 脚本在**基表和动态性能视图及其同义词上创建视图**

还启动其它脚本创建以下项目的对象

1. 基本 PL/SQL 环境包括 PL/SQL 声明

   1)数据类型 2)预定义异常 3)内置过程和函数 4)SQL 操作

2. 审计

3. 导入/导出

4. SQL*Loader

5. 已安装选项

catproc.sql

catproc.sql 脚本**建立 PL/SQL 功能的使用**, 此外它创建几个可用于**扩展RDBMS 功能的 PL/SQL 程序包。**

catproc.sql 脚本还为以下项目创建其它程序包和视图 1 .警报 2 .管道 3 .Logminer 4 .大型对象 5 .对象 6 .高级排队 7 .复制选项 8 .其它内置项目和选项

pupbld.sql

有时出于安全考虑**需要禁止一些业务系统的数据库用户执行sqlplus命令**，方法很简单：在运行命令之前，将这些命令限制到一个由 SQLPlus 引用的"特殊位置"。 此特殊位置是SYSTEM 模式中一个名为**PRODUCT_USER_PROFILE** 的表。 如果该表不存在，则您在每次启动 SQLPlus 时将获得一个类似"Product User Profile Not Loaded"这样的警告。

为了创建这个表，需要运行pupbld.sql脚本。通常，这个脚本在$ORACLE_HOME/sqlplus/admin 路径中运行，具体的位置由系统决定。