## ORACLE数据库管理

目录

**[一、 表管理 5](#_Toc1037)**

[1.1 创建表 5](#_Toc18593)

[1.2 添加列 6](#_Toc11880)

[1.3 修改列定义 6](#_Toc19592)

[1.4 修改列名 6](#_Toc10105)

[1.5 修改表名 6](#_Toc19364)

[1.6 删除列 6](#_Toc344)

[1.7 删除数据 6](#_Toc30224)

[1.8 删除表 6](#_Toc8479)

[1.9 为表加注释 6](#_Toc25660)

[1.10 为列加注释 6](#_Toc28214)

[1.11 改变表的读写模式 7](#_Toc13765)

[1.12 整理表 7](#_Toc20563)

[1.13 高速缓存表 7](#_Toc2027)

[1.14 查看表的所有列信息 7](#_Toc31372)

[1.15 查看表的高水位信息 7](#_Toc2267)

[1.16 索引表 7](#_Toc21511)

**[二、 索引管理 7](#_Toc22056)**

[2.1 创建普通索引 7](#_Toc4868)

[2.2 创建唯一索引 7](#_Toc2086)

[2.3 创建位图索引 7](#_Toc1362)

[2.4 创建反序索引 8](#_Toc6405)

[2.5 创建函数索引 8](#_Toc6522)

[2.6 创建大型索引 8](#_Toc3549)

[2.7 合并与重建索引 8](#_Toc5661)

[2.8 重命名索引 9](#_Toc14607)

[2.9 回收索引未使用的存储空间 9](#_Toc6485)

[2.10 删除索引 9](#_Toc26783)

[2.11 使索引高速缓存 9](#_Toc2080)

**[三、 表空间管理 9](#_Toc10358)**

[3.1 普通表空间 9](#_Toc5656)

[3.2 大文件表空间 9](#_Toc16307)

[3.3 临时表空间 10](#_Toc4109)

[3.4 撤销（UNDO）表空间 11](#_Toc2414)

[3.5 重命名表空间 12](#_Toc10292)

[3.6 更改表空间数据文件 12](#_Toc3220)

[3.7 设置默认表空间 12](#_Toc14602)

[3.8 使表空间在线|离线 12](#_Toc5076)

[3.9 改变表空间读写状态 12](#_Toc23585)

[3.10 删除表空间及数据文件 12](#_Toc19885)

[3.11 查看表空间信息 12](#_Toc23653)

**[四、 表分区管理 13](#_Toc32277)**

[4.1表分区优点 13](#_Toc1964)

[4.2 何时需要表分区 13](#_Toc22113)

[4.3 基本分区方法 14](#_Toc16771)

[4.4 创建范围分区表 14](#_Toc27980)

[4.5 创建列表分区表 15](#_Toc2451)

[4.6 创建散列分区表 15](#_Toc24654)

[4.7 创建复合分区表 15](#_Toc31258)

[4.8 创建间隔分区表 17](#_Toc12458)

[4.9 创建虚拟列分区表 17](#_Toc10063)

[4.10 向有虚拟列的表插值 17](#_Toc16851)

[4.11 添加分区 17](#_Toc20128)

[4.12 删除分区 18](#_Toc19483)

[4.13 交换分区 18](#_Toc32049)

[4.14 合并分区 18](#_Toc20228)

[4.15 拆分分区 18](#_Toc5670)

[4.16 截断分区 18](#_Toc13488)

[4.17 移动分区 19](#_Toc8364)

[4.18 重命名分区 19](#_Toc23451)

[4.19 修改列表分区 19](#_Toc27925)

[4.20 修改子分区模板 19](#_Toc29954)

[4.21 修改分区默认属性 19](#_Toc24221)

[4.22 联合分区（合并散列分区） 19](#_Toc14941)

**[五、 索引分区管理 20](#_Toc18263)**

[5.1 索引分区方法 20](#_Toc16873)

[5.2 本地索引与全局索引区别 20](#_Toc10019)

[5.3 创建本地索引 20](#_Toc3153)

[5.4 创建全局索引 20](#_Toc8994)

[5.5 查询分区表和分区索引信息 21](#_Toc15568)

[XXX : DBA、ALL、USER 21](#_Toc470)

**[六、 视图与实体化视图管理 21](#_Toc14150)**

[6.1 视图与实体化视图定义 21](#_Toc6493)

[6.2 视图的优点 21](#_Toc31766)

[6.3 创建视图所需权限 21](#_Toc18530)

[6.4 创建视图 22](#_Toc6076)

[6.5 视图的DML操作 22](#_Toc27818)

[6.6 修改视图定义 22](#_Toc20944)

[6.7 删除视图 22](#_Toc3906)

[6.8 查询视图信息 22](#_Toc1581)

[6.9 实体化视图的刷新方法 23](#_Toc29144)

[6.10 实体化视图刷新模式 23](#_Toc25351)

[6.11 创建实体化视图所需权限 23](#_Toc17049)

[6.12 创建实体化SQL 23](#_Toc23411)

[6.13 创建手动刷新实体化视图 24](#_Toc14178)

[6.14 创建提交时刷新实体化视图 24](#_Toc2194)

[6.15 创建增量刷新实体化视图 24](#_Toc7353)

[6.16 创建定时刷新实体化视图 24](#_Toc25872)

[6.17 维护实体化视图 24](#_Toc27091)

[6.18 删除实体化视图 24](#_Toc15779)

[6.19 查询视图信息 24](#_Toc27035)

**[七、 用户与权限管理 24](#_Toc18025)**

[7.1 数据库预定义用户 24](#_Toc18704)

[7.2 创建用户 24](#_Toc1112)

[7.3 修改用户 24](#_Toc7543)

[7.4 删除用户 25](#_Toc29477)

[7.5 查询用户 25](#_Toc18736)

[7.6 权限概述 25](#_Toc21310)

[7.7 系统权限授予 25](#_Toc7222)

[7.8 系统权限回收 25](#_Toc10454)

[7.9 对象权限授予 25](#_Toc13709)

[7.10 对象权限回收 25](#_Toc31764)

[7.11 查询权限信息 26](#_Toc5418)

[7.12 创建角色 26](#_Toc11227)

[7.13 角色的授予、回收和删除 26](#_Toc2587)

[7.14 查询角色信息 26](#_Toc28625)

**[八、 常用SQL 26](#_Toc18173)**

[8.1 查看执行计划 26](#_Toc19333)

[8.2 当前正在运行的SQL 27](#_Toc30457)

[8.3 查看锁情况 27](#_Toc1000)

[8.4 解析时间最长的SQL 27](#_Toc16359)

[8.5 查看磁盘读写最多SQL 28](#_Toc29048)

[8.6 查看排序最多SQL 28](#_Toc16966)

[8.7 磁盘争用 28](#_Toc7370)

[8.8 日志切换间隔 28](#_Toc28196)

[8.9 连接数 28](#_Toc16468)

[8.10 创建JOB 28](#_Toc31798)

**附录：**

1、用户账号状态描述（ACOUNT\_STATUS）

2、用户信息描述（DBA\_USERS）

3、数据库信息描述（DATABASE\_PERPERTY）

4、DBA\_PROFILES

5、临时表空间数据文件信息（DBA\_TEMP\_FILES）

6、回退段状态（UNDO\_STATUS）

7、查询表空间信息

8、数据文件、表空间、段、区、块

注：所有[]括起来的，说明可以写，可以不写。

1. **表管理**
   1. **创建表**
      1. 创建标准表

CREATE TABLE 表名 (列名 类型,列名 类型,...) [TABLESPACE 表空间名];

例：CREATE TABLE STUDENT(NAME VARCHAR2(20),AGE NUMBER(2),BIRTHDAY DATE) TABLESPACE USER;

* + 1. 创建事务级表

CREATE GLOBAL TEMPORARY TABLE 表名 (列名 类型,列名 类型,...) ON COMMIT DELETE ROWS [TABLESPACE 表空间名];

例：CREATE GLOBAL TEMPORARY TABLE STUDENT(NAME VARCHAR2(20),AGE NUMBER(2),BIRTHDAY DATE) ON COMMIT DELETE ROWS TABLESPACE USER; 当事务提交时，该表数据清空。

* + 1. 创建会话级表

CREATE GLOBAL TEMPORARY TABLE 表名 (列名 类型,列名 类型,...) ON COMMIT DELETE ROWS [TABLESPACE 表空间名];

例：CREATE GLOBAL TEMPORARY TABLE STUDENT(NAME VARCHAR2(20),AGE NUMBER(2),BIRTHDAY DATE) ON COMMIT PRESERVE ROWS TABLESPACE USER; 当会话断开（结束）时，该表数据清空。

* + 1. 利用其他表创建结构一样的表

CREATE TABLE 表名1 [TABLESPACE 表空间名] AS SELECT \* FROM 表名2;

例：CREATE TABLE STUDENT1 TABLESPACE USER2 AS SELECT \* FROM STUDENT2;

* 1. **添加列**

ALTER TABLE 表名 ADD(列名 类型,列名 类型,...);

例：ALTER TABLE STUDENT ADD(SEX NUMBER(1),HIGH NUMBER(4,1));

* 1. **修改列定义**

ALTER TABLE 表名 MODIFY (列名 类型,列名 类型,...);

例：ALTER TABLE STUDENT MODIFY (AGE NUMBER(3),BIRTHDAY TIMESTAMP(6),...);

* 1. **修改列名**

ALTER TABLE 表名 RENAME COLUMN 旧列名 TO 新列名;

例：ALTER TABLE STUDENT RENAME COLUMN AGE TO EGA;

* 1. **修改表名**

RENAME 旧表名 TO 新表名;

例：RENAME STUDNET TO TEACHER;

* 1. **删除列**

ALTER TABLE 表名 DROP (列名1,列名2,...) [CASCADE CONSTRAINTS];

例：ALTER TABLE STUDENT DROP (SEX,AGE) CASCADE CONSTRAINTS;

注：CASCADE CONSTRAINTS 表示级联删除

* 1. **删除数据**

DELETE FROM 表名 [WHERE ..];--删除部分数据

TRUNCATE TABLE 表名 [REUSE STORAGE];--删除全部数据

例：DELETE FROM STUDENT WHERE AGE>10;

TRUNCATE TABLE STUDENT REUSE STORAGE;

注：TRUNCATE方式不会产生日志，速度很快，但删除的数据不能恢复，适合无用的大表数据删除。

REUSE STORAGE 表示被删除的表的所有存储空间被回收，建议加上。

* 1. **删除表**

DROP TABLE 表名 [CASCADE CONSTRAINTS][PURGE];

例：DROP TABLE STUDENT PURGE;

注：PURGE 表示删除表时，立即清空改变在回收站中的数据。

* 1. **为表加注释**

COMMENT ON TABLE 表名 IS ‘内容’;

例：COMMENT ON TABLE STUDENT IS ‘学生信息表’;

* 1. **为列加注释**

COMMENT ON COLUMN表名.列名 IS ‘内容’;

例：COMMENT ON COLUMN STUDENT.AGE IS ‘学生年龄’;

* 1. **改变表的读写模式**

ALTER TABLE 表名 READ ONLY;--只读

ALTER TABLE 表名 READ WRITE;--读写

* 1. **整理表**

若表经常DML时，可能会产生碎片，通过整理表，消除碎片。

ALTER TABLE MOVE [TABLESPACE 表空间名];

注：如果加表空间，整理表的同时移动表所在的表空间

整理表后，相关索引会失效，要***[重建索引](#重建索引)***。

* 1. **高速缓存表**

当一个表经常使用且不大时，可以将其高速缓存。

ALTER TABLE 表名 STORAGE(BUFFER\_POOL KEEP);--将表高速缓存池

ALTER TABLE POSIMM CACHE;--立即建表数据移动高速缓存

同理将***[索引高速缓存](#索引高速缓存)***。

* 1. **查看表的所有列信息**

DESC表名;

例：DESC POSIMM; DESC DBA\_USERS; DESC DBA\_TEMP\_FILES;

* 1. **查看表的高水位信息**

1. BEGIN DBMS\_STATS.GATHER\_TABLE\_STATS (OWNNAME => '"用户名"',TABNAME => '"表名"',ESTIMATE\_PERCENT => 50); END;--统计表的信息
2. SELECT TABLE\_NAME,NUM\_ROWS,BLOCKS,EMPTY\_BLOCKS FROM USER\_TABLES WHERE TABLE\_NAME='表名';
   1. **索引表**

必须有主键，且数据顺序是按主键形成B树排列的，就相当于一个索引，只是索引的内容就是数据而已。

1. **索引管理**

**2.1 创建普通索引**

CREATE INDEX 索引名 ON 表名(列名1 [DESC|ASC],...) [TABLESPACE 表空间名] [ONLINE];

例：CREATE INDEX STUIDX ON STUDENT(NAME DESC) TABLESPACE USER ONLINE;

注：DESC|ASC 分别表示 降序|升序

ONLINE 表示创建索引时可以进表进行DML操作

**2.2 创建唯一索引**

CREATE UNIQUE INDEX 索引名 ON 表名(列名1[DESC|ASC],...)[TABLESPACE 表空间名][ONLINE];

例：CREATE UNIQUE INDEX STUIDX ON STUDENT(NAME DESC) TABLESPACE USER ONLINE;

**2.3 创建位图索引**

CREATE BITMAP INDEX 索引名 ON 表名(列名1 [DESC|ASC],...) [TABLESPACE 表空间名] [ONLINE];

注：一般用于基数较少的列，如性别，不建议创建位图索引

**2.4 创建反序索引**

CREATE INDEX 索引名 ON 表名(列名1 [DESC|ASC],...) REVRESE [TABLESPACE 表空间名] [ONLINE];

注：反序索引使索引列的字节内容反过来，避免索引字段全部在一个数据块中，提高索引效率，但是该索引失去了范围查找的优点，只能遍历索引。

**2.5 创建函数索引**

CREATE INDEX 索引名 ON 表名(表达式（列名）[DESC|ASC]) [TABLESPACE 表空间名] [ONLINE];

例：CREATE INDEX STUIDX ON STUDENT(SUBSTR(NAME,1,2) DESC) TABLESPACE USER ONLINE;

注：SUBSTR(NAME,1,2) 表示，若NAME=’ABCS’ ,则结果=’AB’。

以NAME字段的这两位做索引。

关键：查询的时候必须使用SUBSTR(NAME,1,2) = ‘内容’，才可以使用该索引

例：SELECT \* FROM STUDENT WHERE SUBSTR(NAME,1,2) = ‘张三’;

**2.6 创建大型索引**

当创建的索引特别大时，建议先创建一个大型的临时表空间，因为要是使用默认的临时表空间的话，会影响数据库其他进程的性能。

步骤：

1. 创建一个新的、较大的临时表空间

CREATE BIGFILE TEMPORARY TABLESPACE 表空间名TEMPFILE 'E:\XX\XX\XX.DBF' SIZE 2G AUTOEXTEND ON NEXT 2G MAXSIZE UNLIMITED;

1. 为操作用户指定新的临时表空间

ALTER USER 用户名TEMPORARY TABLESPACE 临时表空间名;

1. 创建索引
2. 创建索引完成后，将临时表空间恢复成原来的，同步骤2
3. 使新建的表空间离线，为删除此表空间做准备

ALTER TABLESPACE TEMP2 TEMPFILE OFFLINE;

1. 删除新建的临时表空间

DROP TABLESPACE TEMP2 INCLUDING CONTENTS AND DATAFILES;

**2.7 合并与重建索引**

当对表进行删除行操作的时候，只是在索引的叶子节点中逻辑上删除了相应的索引条目，并没有回收空间，更新是也是如此。

注：以上是一些人的说法，测试了业务实时表，索引大小基本不变，所以索引是自动维护的

1. 合并索引

合并叶子节点，将合并后的空结点释放，减小了空间。

ALTER INDEX 索引名 COALESCE;

1. 重建索引

ALTER INDEX 索引名 REBUILD [TABLESPACE 表空间名] [ONLINE];

注：中括号起了移动索引表空间的作用。使用ONLINE，重建索引时，不会影响当前索引的使用。

重建索引需要额外的临时表空间，是先删除索引，再创建索引需要的内存的2倍。

通常是优先考虑index coalesce（索引合并），而不是重建索引。

1. 合并索引优点

1、不需要占用近磁盘存储空间 2 倍的空间；

2、可以在线操作；

3、无需重建索引结构，而是尽快地合并索引叶块，这样可避免系统开销过大。

1. 真正需要重建索引的情形

索引分区失效时，需要重建。

正常情况下可以先删除索引，再重新创建索引。当有许多业务需要用到该索引时，

就要用在线索引重建了(REBUILD ONLINE)。

**2.8 重命名索引**

ALTER INDEX 旧索引名 RENAME TO 新索引名;

**2.9 回收索引未使用的存储空间**

ALTER INDEX 索引名 DEALLOCATE UNUSED [KEEP 大小];

注：中括号 表示使索引收缩到指定大小

**2.10 删除索引**

DROP INDEX 索引名;

**2.11 使****索引高速缓存**

引当经常使用某个小索引时，可以像表高速缓存一样，将索引高速缓存。

ALTER INDEX索引名 STORAGE(BUFFER\_POOL KEEP);

1. **表空间管理**

**3.1 普通表空间**

1）创建普通表空间

CREATE TABLESPACE 表空间名DATAFILE

'E:\XX\XX\XX\XXX.DBF' SIZE 10M [AUTOEXTEND ON NEXT 5M MAXSIZE 100M],

'E:\XX\XX\XX\XXX.DBF' SIZE 10M [AUTOEXTEND ON NEXT 5M MAXSIZE 100M];

注：一个表空间可以对应多个数据文件，创建完之后，还可以给这表空间添加数据文件。

中括号里 表示自动扩展，每次扩展5M，最大大小为100M 。最大大小可以设成无限(UNLIMITED)。

但普通表空间单个数据文件最大是32G，所以UNLIMITED表示32G。

2）添加普通表空间数据文件

ALTER TABLESPACE 表空间名ADD DATAFILE 'E:\ORACLE\VITS\XXX.DBF' SIZE 530M [AUTOEXTEND ON NEXT 50M MAXSIZE 2G];

3）删除普通表空间数据文件

ALTER TABLESPACE 表空间名DROP DATAFILE 'E:\ORACLE\VITS\DATA\TEST01 ';

不建议创建普通表空间，维护麻烦。

**3.2 大文件表空间**

大文件表空间只有一个数据文件，数据文件理论上无限大，因为肯定比磁盘大。

1）创建大文件表空间

CREATE BIGFILE TABLESPACE 表空间名DATAFILE '路径' SIZE 100M AUTOEXTEND ON NEXT 100M MAXSIZE UNLIMITED;

1. 因为大文件表空间就一个数据文件，所以没有添加|删除数据文件的操作，只能直接删除表空间。

**3.3 临时表空间**

1、临时表空间用来管理数据库排序操作以及用于存储临时表、中间排序结果等临时对象,当ORACLE里需要用到SORT的时候，并且当PGA中sort\_area\_size大小不够时，将会把数据放入临时表空间里进行排序。

2、像数据库中一些操作： CREATE INDEX、 ANALYZE、SELECT DISTINCT、ORDER BY、GROUP BY、 UNION ALL、 INTERSECT、MINUS、SORT-MERGE JOINS、HASH JOIN等都可能会用到临时表空间。

3、当操作完成后，系统会自动清理临时表空间中的临时对象，自动释放临时段。这里的释放只是标记为空闲、可以重用，其实实质占用的磁盘空间并没有真正释放。这也是临时表空间有时会不断增大的原因。

4、如果有必要，可以为一些特殊的用户指定特定的表空间。

5、合理设置PGA，减少临时表空间使用的几率。

6、要为临时表空间保留足够的硬盘空间。

7、***[临时表空间过大怎么办](#表空间过大解决方法)***。

Union，对两个结果集进行并集操作，不包括重复行，同时进行默认规则的排序；

Union All，对两个结果集进行并集操作，包括重复行，不进行排序；

Intersect，对两个结果集进行交集操作，不包括重复行，同时进行默认规则的排序；

Minus，对两个结果集进行差操作，不包括重复行，同时进行默认规则的排序。

可以在最后一个结果集中指定Order by子句改变排序方式。

1）创建临时表空间

CREATE [BIGFILE] TEMPORARY TABLESPACE 表空间名 TEMPFILE '路径' SIZE 10M AUTOEXTEND ON NEXT 10M MAXSIZE UNLIMITED;

2）添加临时表空间数据文件

ALTER TABLESPACE 表空间名ADD TEMPFILE '路径' SIZE 100M AUTOEXTEND ON NEXT 50M MAXSIZE 5G;

3）删除临时表空间数据文件

ALTER TABLESPACE 表空间名 DROP TEMPFILE ‘路径’;

4）收缩临时表空间

ALTER TABLESPACE 表空间名SHRINK SPACE [KEEP 20M];

5）收缩表空间不一定成功，因为有其他线程占用，最稳健的方法是先创建一个新的临时表空间，在将此表空间设置成默认临时表空间，把旧的表空间删除。（**表空间过大解决方法**）

6）查看默认临时表空间

1. SELECT \* FROM ***[DATABASE\_PROPERTIES](#数据库信息描述)*** WHERE PROPERTY\_NAME='DEFAULT\_TEMP\_TABLESPACE';--数据库默认临时表空间
2. SELECT TEMPORARY\_TABLESPACE FROM ***[DBA\_USERS](#用户信息描述)*** WHERE USERNAME='用户名';--用户默认临时表空间，用户名要大写！
3. SELECT USERNAME,***[ACCOUNT\_STATUS](#用户账号状态描述)***,DEFAULT\_TABLESPACE,TEMPORARY\_TABLESPACE FROM DBA\_USERS;--11G开始不显示密码和一些其他信息了！

7）指定自定义的默认临时表空间

1. ALTER DATABASE DEFAULT TEMPORARY TABLESPACE 表空间名;--设置数据库的默认临时表空间
2. ALTER USER 用户名TEMPORARY TABLESPACE 表空间名;--设置单个用户的默认临时表空间

8）查看临时表空间信息

SELECT TABLESPACE\_NAME,STATUS,MAXBYTES/1024/1024/1024 MAXSIZE\_G,BYTES/1024/1024/1024 REALSIZE\_G,USER\_BYTES/1024/1024/1024 USEDSIZE\_G,FILE\_NAME FROM ***[DBA\_TEMP\_FILES](#临时表空间数据文件信息)***;

9）删除临时表空间

1、ALTER TABLESPACE 表空间名 TEMPFILE OFFLINE;

2、DROP TABLESPACE 表空间名 INCLUDING CONTENTS AND DATAFILES;

10）查看临时表空间组

SELECT \* FROM DBA\_TABLESPACE\_GROUPS;

1. 添加临时表空间组

ALTER TABLESPACE 表空间名TABLESPACE GROUP GROUP1;

1. 设置默认临时表空间组

1、ALTER USER 用户名TEMPORARY TABLESPACE GROUP1;--用户默认表空间组

2、ALTER DATABASE DEFAULT TEMPORARY TABLESPACE GROUP1;--数据库默认表空间组

1. 退出临时表空间组

ALTER TABLESPACE 表空间名TABLESPACE GROUP '';

1. 临时表空间组的作用
2. 避免由于单个临时表空间的空间不足而导致数据库运行故障。
3. 保证一个并行操作中的多个并行任务在多个临时表空间中执行。

**3.4 撤销（UNDO）表空间**

1、当某个事物修改数据时，ORACLE首先将数据的原始值保持到一个回退段中。一个事务只能将它的全部修改数据的原始值保存到一个回退段中。多个事务可以共用一个回退段。利用

2、当一个事务开始时，系统分配给该事务一个回退段，当数据发生改变时，数据的原始值才被复制到回退段中。回退段由区组成，区由数据块组成。与缓冲区类似，采用循环写的方式工作，当所有区都写满后，事务开始循环写入第一个区或给回退段分配新的区（若设置了自动扩展的话）。

3、回退段中保存的回退信息，可以实现事务回滚、数据库恢复、数据读一致性和闪回查询。

而重做日志信息作用是，当由于某些原因导致事务对数据的修改（已提交）丢失时，进行恢复。当事务提交时，系统将重做日志缓冲区中的重做信息写入磁盘中的重做日志文件，永久保存。重做日志文件采用多路复用形式，有多个副本。

4、如果已提交事务对数据的修改结果还没有写入数据文件中，发生系统崩溃，那么系统重启时，将会根据重做日志文件重新实现对已提交事务对数据的修改，以及利用回退信息取消对未提交事务对数据的修改。

1. 事务回滚

当事务执行失败或用户执行事务回滚操作时，ORACLE会利用保存在回退段中的数据的原始值将数据恢复到原来的值。

1. 数据库恢复

当数据库实例运行失败时，在数据库恢复时，ORACLE先利用重做日志文件中的信息对数据

文件进行恢复（包括提交的事务和未提交的事务），然后利用回退段中的信息回滚未提交事务对数据的修改。

1. 读一致性

当一个用户对数据库进行修改，但还没提交时，系统将用户修改的数据的原始值保存在回退段中，这样就可以为正在访问相同数据的其他用户提供了一份该数据的原始副本，从而使得当前用户未提交的修改其他用户无法看到，保证了数据的读一致性。

1. 闪回查询

该技术可以查询某个数据在过去某个时间点的状态。闪回查询技术是利用回退段中保存的数据原始信息实现的。

回退段中保存的事务回退信息时间的长短取决于数据库事务量的多少、回退段空间大小以及数据库的配置情况。

1. 创建撤销表空间

CREATE [BIGFILE] UNDO TABLESPACE 表空间名 DATAFILE '路径' SIZE 10M AUTOEXTEND ON NEXT 10M MAXSIZE UNLIMITED;

1. 可以创建多个撤销表空间，但只能有一个处于活动状态
2. 回退信息保留时间

1、UNDO\_RETENTION：回退信息保留的最短时间

2、若撤销表空间大小固定，该参数被忽略，数据库自动调节保留时间，力争时间最长

3、如果表空间已经扩展到最大，将覆盖已提交的回退信息，即使回退信息还没有过期，这是会产生ORA-01555错误

4、可以设置RETENTION GUARANTEE特性，保证只有过期的回退信息会被覆盖，即使表空间空间不足，但数据库会因为没有可用的UNDO空间而挂起

1）ALTER SYSTEM SET UNDO\_RETENTION = 1800;--设置保留时间，单位S

2）ALTER TABLESPACE 撤销表空间名 RETENTION GUARANTEE|NOGUARANTEE;--设置GUARANTEE特性

3）SELECT TABLESPACE\_NAME, RETENTION FROM DBA\_TABLESPACES WHERE TABLESPACE\_NAME='UNDOTBS2';

1. 回退段信息
2. SELECT TABLESPACE\_NAME,SEGMENT\_NAME,***[STATUS](#回退段状态)***,BYTES/1024/1024 SIZE\_M FROM ***DBA\_UNDO\_EXTENTS;***
3. 切换撤销表空间

1、ALTER SYSTEM SET UNDO\_TABLESPACE = UNDOTBS2;--重新指定新创建的表空间

2、ALTER DATATABLE UNDOTBS1 OFFLINE;--只有等待此表空间中所有回退段都不是ACTIVE时，才能成功使其脱机。

3、DROP TABLESPACE UNDOTBS1 INCLUDING CONTENTS AND DATAFILES;

**3.5 重命名表空间**

ALTER TABLSPACE 旧表空间名 RENAME TO 新表空间名;

**3.6 更改表空间数据文件**

ALTER DATABASE DATAFILE 'E:\ORACLE\VITS\DATA3\POSHIS06\_2.DBF' [RESIZE 200M]| [AUTOEXTEND ON NEXT 100M MAXSIZE 2G];

注：| 表示 每次最多指定一个条件。

**3.7 设置默认表空间**

1、ALTER USER 用户名 DEFAULT TABLESPACE 表空间名;--单个用户默认表空间

2、ALTER DATABASE DEFAULT TABLESPACE 表空间名;--数据库级别默认表空间

**3.8 使表空间在线|离线**

1、ALTER TABLESPACE 表空间名 OFFLINE|ONLINE; --普通表空间，UNDO表空间

2、ALTER TABLESPACE 表空间名 TEMPFILE OFFLINE|ONLINE;--临时表空间

**3.9 改变表空间读写状态**

ALTER TABLESPACE 表空间名 READ ONLY|READ WRITE;

**3.10 删除表空间及数据文件**

1、ALTER TABLESPACE 表空间名 OFFLINE; --先离线再删

2、DROP TABLESPACE 表空间名 INCLUDING CONTENTS AND DATAFILES;

**3.11 查看表空间信息**

1、查看表空间及对应数据文件

SELECT V2.NAME,V1.NAME FROM V$DATAFILE V1,V$TABLESPACE V2 WHERE V1.TS#=V2.TS#(+) ORDER BY V2.NAME;--普通表空间

SELECT V2.NAME,V1.NAME FROM V$TEMPFILE V1,V$TABLESPACE V2 WHERE V1.TS#=V2.TS#(+) ORDER BY V2.NAME;--临时表空间

2、查看表空间大小信息

SELECT TABLESPACE\_NAME,STATUS,MAXBYTES/1024/1024/1024 MAXSIZE\_G,BYTES/1024/1024/1024 REALSIZE\_G,USER\_BYTES/1024/1024/1024 USEDSIZE\_G,FILE\_NAME FROM DBA\_DATA\_FILES;--普通表空间

SELECT TABLESPACE\_NAME,STATUS,MAXBYTES/1024/1024/1024 MAXSIZE\_G,BYTES/1024/1024/1024 REALSIZE\_G,USER\_BYTES/1024/1024/1024 USEDSIZE\_G,FILE\_NAME FROM ***[DBA\_TEMP\_FILES](#临时表空间数据文件信息)***;--临时表空间

3、查看表空间状态

SELECT TABLESPACE\_NAME,STATUS,LOGGING,BIGFILE FROM DBA\_TABLESPACES;--有LOGGING

SELECT V2.NAME,V1.NAME,V1.ENABLED,STATUS,BIGFILE FROM V$DATAFILE|V$TEMPFILE V1,V$TABLESPACE V2 WHERE V1.TS#=V2.TS#(+) ORDER BY V2.NAME;--有数据文件名和读写状态

4、查看表空间包含的段信息（数据段，索引段）

SELECT OWNER,SEGMENT\_NAME,SEGMENT\_TYPE,PARTITION\_NAME,TABLESPACE\_NAME,BUFFER\_POOL,BYTES/1024/1024 SIZE\_M,MAX\_SIZE/1024/1024/1024 MAXSIZE\_G FROM DBA\_SEGMENTS WHERE TABLESPACE\_NAME=表空间名 [AND SEGMENT\_NAME = 表名];

1. **表分区管理**

**4.1表分区优点**

1. 提高数据库的可管理性。通过对表进行分区，可以使数据的增删改、重建索引、备份与恢复等操作在分区级别上进行，而不需要在整个表的级别进行。
2. 增强数据库的可用性。如果分区出现故障，只影响该分区的使用，而对表或索引的其他分区的使用不受影响。
3. 简化数据库的维护。如果表或索引的某个分区出现故障，只需对该分区进行数据恢复操作。
4. 改善查询性能。
5. 提高数据库操作的并行性。对表或索引分区后，可以对表或索引进行并行操作，包括查询操作、DML操作、DDL操作等，减少执行时间，优化了资源利用。
6. 透明性，分区操作，对应用而言是透明的。

**4.2 何时需要表分区**

1）如果表大小超过2GB。

2）如果要对表进行并行DML操作时。

3）为了平衡硬盘的I/O操作，需要将一个表分散存储到不用的表空间时。

4）如果需要将表的一部分设置为只读状态，另一部分为读写状态时。

5）如果需要将表的一部分设置成联机状态，另一部分为脱机状态是。

注：如果表中包含LONG或LONG RAW类型类时，不能对表进行分区。

**4.3 基本分区方法**

1）范围分区：根据分区列的值的范围对表进行分区。

2）列表分区：分区列的取值是少数值的集合

3）散列分区：采用基于分区列值的HASH算法，将数据均匀分布到指定的分区中。避免数据不均衡的情况。

4）复合分区：结合两个基本分区，先采用一个分区对表或索引进行分区，让后再用另一个对每个分区分成若干个子分区。

5）间隔分区：是范围分区的一种扩展，当向表中插入的数据的分区列值不在所有存在的分区范围是，会自动创建特定间隔的分区，必须至少指定一个范围分区，称为转换点，数据库将为超过该转换点的数据创建间隔分区。

6）引用分区：使子表继承父表的分区方法

7）基于虚拟列分区：与基于普通列分区完全相同

8）系统分区：分区由系统来控制

Oracle支持的复合分区：

1. 范围-范围 2、范围-散列 3、范围-列表
2. 列表-范围 5、列表-散列 6、列表-列表

7、间隔-范围 8、间隔-散列 9、间隔-列表

间隔分区注意事项：

1. 只能指定一个分区列，并且必须是NUMBER类型或DATE类型。
2. 索引表不支持间隔分区。
3. 不能为间隔分区表创建域索引（oracle词法分析器的全文索引）。

**4.4 创建范围分区表**

1、时间范围

CREATE TABLE 表名 (...) [TABLESPACE XXX] PARTITION BY RANGE (列1[,列2])

(

PARTITION 分区名 VALUES LESS THAN(TO\_TIMESTAMP(’2018-05-01 00:00:00’,’YYYY-MM-DD HH24:MI:SS’)) [TABLESPACE XXX]

[,PARTITION 分区名 VALUES LESS THAN(TO\_TIMESTAMP(‘2018-06-01 00:00:00’,’YYYY-MM-DD HH24:MI:SS’)) [TABLESPACE XXX]]

,PARTITION 分区名 VALUES LESS THAN(MAXVALUE) [TABLESPACE XXX]

) ;

2、数字范围

CREATE TABLE 表名 (...) [TABLESPACE XXX] PARTITION BY RANGE (列1[,列2])

(

PARTITION 分区名 VALUES LESS THAN(100) [TABLESPACE XXX]

[,PARTITION 分区名 VALUES LESS THAN(1000) [TABLESPACE XXX]]

,PARTITION 分区名 VALUES LESS THAN(MAXVALUE) [TABLESPACE XXX]

) ;

**4.5 创建列表分区表**

CREATE TABLE 表名 (...) [TABLESPACE XXX] PARTITION BY LIST(列名)

(

PARTITION 分区名 VALUES ([literal|null]|[DEFAULT]) [TABLESPACE XXX]

[,PARTITION 分区名 VALUES ([literal|null]|[DEFAULT]) [TABLESPACE XXX]]

) ;

例：

CREATE TABLE STUDENT (NAME VARCHAR2(20),AGE NUMBER(2)) TABLESPACE USER PARTITION BY LIST(NAME)

(

PARTITION P1 VALUES (‘张三’,’李四’ ) TABLESPACE USER

,PARTITION P2 VALUES(NULL) TABLESPACE POSIMM

,PARTITION P3 VALUES(DEFAULT) TABLESPACE IPOSIMM

)

注：VALUES中的值可以是NULL，可以是特定是值如：’东南’，若是DEFALUT的话，则说明不符合其他分区的要求，都保存在该分区中。列表分区值能根据一个列来分区。

**4.6 创建散列分区表**

CREATE TABLE 表名 (...) [TABLESPACE XXX] PARTITION BY HASH(列名1[,列名2])

[

(

PARTITION 分区名 [TABLESPACE XXX]

[,PARTITION 分区名 [TABLESPACE XXX]]

)

]

|

[

PARTITIONS 分区数 STORE IN(T1 [,T2])

]

;

例：

CREATE TABLE STUDENT (NAME VARCHAR2(20),AGE NUMBER(2)) TABLESPACE USER PARTITION BY HASH(NAME)

PARTITIONS 4 STORE IN (T1,T2,T3,T4);

注：两个分区的方法只能选一个T1,T2是表空间名

**4.7 创建复合分区表**

1）范围-列表

CREATE TABLE STUDENT (BIRT DATE,NAME VARCHAR2(22))

PARTITION BY RANGE(BIRT)

SUBPARTITION BY LIST(NAME)

(

PARTITION P1 VALUES LESS THAN(TO\_DATE(‘2018-05-01’,’YYYY-MM-DD’))

(

SUBPATITION P21 VALUES(‘WK’,’Z’)

,SUBPARTITION P22 VALUES(DEFALUT)

),

PARTITION P2 VALUES LESS THAN(TO\_DATE(‘2018-06-01’,’YYYY-MM-DD’))

(

SUBPATITION P21 VALUES(‘WK’,’Z’)

,SUBPARTITION P22 VALUES(DEFALUT)

),

PARTITION P3 VALUES LESS THAN(MAXVALUE)

(

SUBPATITION P31 VALUES(‘WK’,’Z’)

,SUBPARTITION P32 VALUES(DEFALUT)

,SUBPARTITION P33 VALUES(NULL)

)

)

2）范围-散列

CREATE TABLE STUDENT (BIRT DATE,NAME VARCHAR2(22))

PARTITION BY RANGE(BIRT)

SUBPARTITION BY HASH(NAME)

SUBPARTITIONS 4 STORE IN (T1,T2,T3,T4)

(

PARTITION P1 VALUES LESS THAN(TO\_DATE(‘2018-05-01’,’YYYY-MM-DD’)),

PARTITION P2 VALUES LESS THAN(TO\_DATE(‘2018-06-01’,’YYYY-MM-DD’))

)

1. 范围-范围

CREATE TABLE STUDENT (BIRT DATE,AGE NUMBER(2))

PARTITION BY RANGE(BIRT)

SUBPARTITION BY RANGE(AGE)

(

PARTITION P1 VALUES LESS THAN(TO\_DATE(‘2018-05-01’,’YYYY-MM-DD’))

(

SUBPATITION P11 VALUES LESS THAN(14) TABLESAPCE T1

,SUBPARTITION P12 VALUES LESS THAN(MAXVALUE) TABLESAPCE T2

)

,

PARTITION P2 VALUES LESS THAN(TO\_DATE(‘2018-06-01’,’YYYY-MM-DD’))

(

SUBPATITION P11 VALUES LESS THAN(14)

,SUBPARTITION P12 VALUES LESS THAN(MAXVALUE)

)

)

**4.8 创建间隔分区表**

CREATE TABLE STUDENT (BIRT DATE,NAME VARCHAR2(12))

PARTITION BY RANGE(BIRT)

INTERVAL(NUMTOMINTERVAL(1,’MONTH’)) [STORE IN (T1,T2,T3..)]

(

PARTITION P1 VALUES LESS THAN (TO\_DATE(‘2018-07-01’,’YYYY-MM-DD’)) [TABLESPACE T4]

[,PARTITION P2 VALUES LESS THAN (TO\_DATE(‘2018-08-01’,’YYYY-MM-DD’)) [TABLESPACE T4]]

)

注：此表是按月来分区的，当有超过2018-08-01的时间是，自动创建相应分区

**4.9 创建虚拟列分区表**

CREATE TABLE STUDENT(CO1 NUMBER(2),CO2 NUMBER(2),CO\_VIRTUL AS (CO1\*CO2))

PARTITION BY RANGE(CO\_VIRTUL)

(

PARTITION P1 VALUES LESS THAN(10)

,PARTITION P2 VALUES LESS THAN(100)

,PARTITION P3 VALUES LESS THAN(MAXVALUE)

)

**4.10 向有虚拟列的表插值**

INSERT INTO STUDENT VALUES(1,2); --这样提示值不足

INSERT INTO STUDENT (CO1,CO2) VALUES(1,2); --这样虚拟列才有值

INSERT INTO STUDENT (CO1) VALUES(1); --这样虚拟列的值时NULL

**4.11 添加分区**

ALTER TABLE XXX ADD PARTITION ...;

ALTER TABLE XXX MODIFY PARTITION XXX ADD SUBPARTITION...;--复合分区添加子分区

例：

ALTER TABLE STUDENT ADD PARTITION P3 VALUES LESS THAN(TO\_DATE(‘2018-09-02’,’YYYY-MM-DD’))

(

SUBPARTITION P31 VALUES LESS THAN(20),

SUBPARTITION P32 VALUES LESS THAN(MAXVALUE)

)

注：若添加的是范围分区，添加的分区范围必须大于最大的分区值，所以一般用不能用添加分区方式添加分区，而只能通过***拆分分区***方式添加分区。

**4.12 删除分区**

ALTER TABLE XXX DROP PARTITION XXX;

ALTER TABLE XXX DROP SUBPARTITION XXX;

注：

1. 如果要删除一个散列分区或散列子分区，只能采用***联合分区***的方法进行操作。
2. 不能删除引用分区，如果删除父表中的分区，所有子表的引用分区也级联删除。
3. 如果要保留分区中的数据，应该采用合并分区的方式删除分区。

删除表的分区时，自动删除分区对应的本地分区索引，且所有全局索引或全局分区索引，都变成UNUSABLE状态，需要重建，对本地索引没影响。

**4.13 交换分区**

是对分区表与非分区表、分区与复合分区（两个表，分区和复合分区对应的条件必须要满足）之间进行数据的交换。

ALTER TABLE 分区表 EXCHANGE PARTITION XXX WITH TABLE 非分区表 [INCLUDING INDEXES] [WITHOUT VALIDATION];

注：

1. 如果指定INCLUDING INDEXES表示同时交换本地分区索引
2. 指定WITHOUT VALIDATION表示，不进行数据的合法性验证，但如果有主键或唯一性约束，则不起作用。

**4.14** **合并分区**

合并分区的实质是新建一个分区或子分区，保存原来俩个分区或子分区的数据，然后删除原来的两个分区或子分区。

ALTER TABLE XXX MERGE PARTITIONS P1,P2 INTO PARTION P1|P2;

注：

1. 不能使用MERGE语句合并散列分区或散列子分区，只能使用***[联合分区](#联合分区)***合并。
2. 如果合并的是范围分区或子分区，必须是相邻的两个分区，合并后分区名字只能是边界较大的或新的名字，且分区值较大的边界，是新分区的边界。
3. 不能合并引用分区，在合并父表分区的同时会级联合并子表中的引用分区。

**4.15 拆分分区**

如果一个分区中的数据过多，为了提高数据管理的效率，减少磁盘I/O，可以将一个分区拆成两个分区。

ALTER TABLE XX SPLIT PARTITION P1 AT|VALUES(...) INTO (PARTITION..);

例：

ALTER TABLE STUDENT SPLIT PARTITION P1 AT(TO\_DATE(‘’2018-08-01,’YYYY-MM-DD’)) INTO (PARTITION P11,PARTITION P12);

ALTER TABLE STUDENT SPLIT SUBPARTITION P11 VALUES(‘A’,’B’) INTO (SUBPARTITION P111,SUBPARTITION P112);

**4.16 截断分区**

删除某个分区的数据，但不删除分区，即使分区表只有一个分区，也可以截断该分区。

ALTER TABLE XXX TRUNCATE PARTITION XXX;

ALTER TABLE XXX TRUNCATE SUBPARTITION XXX;

**4.17 移动分区**

移动分区可以减少存储碎片、修改分区或子分区创建时的属性设置、进行表中数据压缩以及将分区移动到一个新的表空间等。

ALTER TABLE XXX MOVE PARTITION XXX [TABLESPACE XXX] [COMPRESS] [NOLOGGING];

ALTER TABLE XXX MOVE SUBPARTITION XXX [TABLESPACE XXX] [COMPRESS] [NOLOGGING];

**4.18 重命名分区**

ALTER TABLE XXX RENAME PARTITION OLDP TO NEWP;

ALTER TABLE XXX RENAME SUBPARTITION OLDP TO NEWP;

**4.19 修改列表分区**

1）添加值

ALTER TABLE XXX MODIFY PARTITION XX ADD VALUES(..);

ALTER TABLE XXX MODIFY SUBPARTITION XX ADD VALUES(..);

注：为列表分区添加的值不能在其他分区中出现，不能为DEFAULT分区添加值。

1. 删除值

ALTER TABLE XX MODIFY PARTITION XX DROP VALUES(..);

ALTER TABLE XX MODIFY SUBPARTITION XX DROP VALUES(..);

注：如果分区表中存在与要删除的分区列值相关的记录，则Oracle会返回错误，删除操作无法完成，可以先试用DELETE语句删除相关记录，再进行。

**4.20 修改子分区模板**

在将某个分区表删掉并重建之后发现一个问题，就是现在新建分区之后，自动创建的子分区格式与以前不同。

修改默认子分区模板之后，创建新的分区就会自动生成符合格式要求的子分区。

ALTER TABLE STUDENT

SET SUBPARTITION TEMPLATE (

SUBPARTITION SP\_00 VALUES (0),SUBPARTITION SP\_01 VALUES (1),SUBPARTITION SP\_02 VALUES (2),

);

**4.21 修改分区默认属性**

修改结果只对以后创建的分区有效。

ALTER TABLE XX MODIFY DEFAULT ATTRIBUTES ...;

ALTER TABLE XX MODIFY DEFAULT ATTRIBUTES FOR PARTITION ...;

例：ALTER TABLE STUDENT MODIFY DEFAULT ATTRIBUTES TABLESPACE TBS\_CDR;

**4.22** **联合分区（合并散列分区）**

将一个散列分区中的数据移植到其他散列分区中，然后删除该分区，即减少一个散列分区，但数据没丢失，同***[合并分区](#合并分区)***。

ALTER TABLE XX COALESCE PARTITION...;

ALTER TABLE XX MODIFY PARTITION XX COALESCE SUBPARTITION;

例：

ALTER TABLE STUDENT COALESCE PARTITION P4; --删除p4分区，并将数据移到其他分区中。

ALTER TABLE STUDENT MODIFY PARTITION P1 COALESCE SUBPARTITION p41;--删除p41子分区，并将数据移到其他子分区中。

1. **索引分区管理**

与表类似，可以对大型索引进行分区，以提高索引的可管理性、可用性以及执行效率。

**5.1 索引分区方法**

1. 本地分区索引

为表的各个分区单独创建索引分区，各个索引分区之间是相互独立的，索引的分区与表的分区是一一对应的，如果为分区添加新的分区，Oracle会自动为新分区建立新的索引分区。由于本地分区索引易于管理，因此，在允许的情况下应尽量为分区表创建本地分区索引。

优点：

1. 如果分区表只有一个分区需要维护，则只有一个索引分区受到影响。
2. 本地分区索引支持分区的独立性，可以独立创建某个索引分区。
3. 本地分区索引支持单一分区数据的装入和卸出。
4. 本地分区索引采用与分区表相同的分区方法，有利于生成高效的查询计划。
5. 本地分区索引简化了表空间的不完全恢复。使用本地分区索引，可以在恢复表分区或子分区的同时恢复其对应的索引分区。
6. 位图分区索引只能用本地分区索引。
7. 全局分区索引

对整个表建立索引，然后再对索引进行分区。索引的分区之间不是相互独立的，索引分区与表分区之间也不是一一对应的。

1. 全局非分区索引

即普通索引

**5.2 本地索引与全局索引区别**

1、本地索引可以以分区列开头（前缀局部索引），也可以一其他列开头（非前缀局部索引）；

2、全局索引只能是前缀局部索引。

3、若是非前缀局部索引，查询的时候带上分区列，就可以很好的利用索引了。

4、本地索引只支持分区内的唯一性，无法支持表上的唯一性，因此如果要用本地索引给表做唯一性索引约束，则索引中必须包括分区列。

**5.3 创建本地索引**

**本地索引仅仅是对每个分区创建索引，索引列为任意列，但为了快速查询，查询条件一定要保护分区列。要不然就是全分区扫描了。**

CREATE INDEX STUIDX ON STUDENT(BIRT) LOCAL

[

(

PARTITION P1 [TABLESPACE XX]

[,PARTITION P2 [TABLESPACE XX]]

)

]

**5.4 创建全局索引**

分区列的主列，必须是索引列的主列

CREATE INDEX STUIDX ON STUDENT(BIRT) GLOBAL PARTITION BY RANGE(BIRT)

(

PARTITION P1 VALUES LESS THAN(TO\_DATE(‘2018-02-01’,’YYYY-MM-DD’))

[,PARTITION P2 VALUES LESS THAN(TO\_DATE(‘2018-02-01’,’YYYY-MM-DD’))]

,PARTITION PMAX VALUES LESS THAN(MAXVALUE)

);

**5.5 查询分区表和分区索引信息**

**XXX : DBA、ALL、USER**

|  |  |
| --- | --- |
| **Column** | **Description** |
| XXX\_PART\_TABLES | 包含分区表的分区信息 |
| XXX\_TAB\_PARTITIONS | 包含分区层次、分区存储、分区统计等信息 |
| XXX\_TAB\_SUBPARTITIONS | 包含子分区的同上信息 |
| XXX\_PART\_KEY\_COLUMNS | 包含分区表的分区列信息 |
| XXX\_SUBPART\_KEY\_COLUMNS | 包含复合分区表的子分区列信息 |
| XXX\_PART\_COL\_STATISTICS | 包含分区表的分区列的统计信息和柱状图信息 |
| XXX\_SUBPART\_COL\_STATISTICS | 包含复合分区表的同上信息 |
| XXX\_PART\_HISTOGRAMS | 包含分区表的柱状图信息 |
| XXX\_SUBPART\_HISTOGRAMS | 子分区表同上信息 |
| XXX\_PART\_INDEXES | 包含分区索引的分区信息 |
| XXX\_IND\_PARTITIONS | 包含分区索引的分层次、分区储存、分区统计等信息 |
| XXX\_IND\_SUBPARTITIONS | 同上 |
| XXX\_SUBPARTITION\_TEMPLATES | 包含存在的分区索引的分区模板信息 |

1. **视图与实体化视图管理**

**6.1 视图与实体化视图定义**

视图：是为了简化数据查询、提高数据安全性而创建的虚表，视图在数据库中只有定义，不占存储空间。用户对视图的操作直接转换为对基表的操作。

实体化视图：是保存远程数据在本地副本的实表，占存储空间，需要不断刷新，才能保持实体化视图与基表数据的一致性。

**6.2 视图的优点**

1）可以像对表一样对视图进行操作，包括视图查询、数据更新、数据插入、数据删除等，系统会自动将对视图的操作转换为对基表的操作。

2）由于视图可以显示表中的选定列，因而可以限制对数据的访问。

3）对于较复杂的SELECT语句的查询，可以使用视图来降低执行这些查询的复杂度。

4）可以使用视图从多个表中检索数据，从而为用户提高数据的独立性。

5）在基表与应用程序之间多了一个中间环节，提高了数据的安全性。

**6.3 创建视图所需权限**

1）在自己模式中创建视图，用户需要CREATE VIEW系统权限。

2）在其他模式中创建视图，用户需要CREATE ANY VIEW系统权限。

3）用户可以直接获得或通过角色间接获得这些权限。

4）视图所有者必须显示获得视图中的对象（表等）的操作权限，不能通过角色间接获得。

5）对视图的操作权限，取决于对视图中对象（表等）的操作权限，如过对基表只有INSERT权限，那么只能对视图进行INSERT操作。

6）如果要将自己的视图赋予给其他用户，其他用户还可以将此视图赋予另外用户，必须还要赋予传递性权限，即授权时使用WITH GRANT OPTION 或 WITH ADMIN OPTION。同理表或其他对象的传递性权限。

**6.4 创建视图**

CREATE [OR REPLACE] [FORCE|NOFORCE] VIEW XX [(字段别名[,字段别名])] AS [具体操作] [WITH CHECK OPTION][WITH READ ONLY];

注：

1）FORCE：不管基表是否存在都创建视图，若基表不存在时，报错，视图创建成功，但失效，一旦基表建立时，视图立即生效。

2）NOFORCE：仅当基表存在时，才能创建成功（默认）。

3）WITH READ ONLY：指明该视图为只读视图，不能进入DML操作。

4）WITH CHECK OPTION：只针对对视图的INSERT和UPDATE操作，当视图有WHERE条件时，对视图进行插入的数据和修改后的数据必须满足WHERE条件，否则失败。

例：CREATE VIEW SS AS SELECT \* FROM STUDENT WHERE AGE>10 WITH CHECK OPTION;

INSERT INTO SS VALUES(9);--操作失败

UPDATE SS SET AGE = 1 WHERE AGE = 11;--操作失败

**6.5 视图的DML操作**

1）对于简单视图，如果视图中包含了基表的主键列已经所有非空列，则可以进行DML操作。

2）对应复杂视图，不能直接进行DML操作，需要使用触发器进行处理。

复杂视图包括：

1. 使用组函数（SUM、AVG等）
2. 使用GROUP BY 子句进行分组
3. 使用DISTINCT关键字
4. 使用ROWNUM伪列
5. 包含表达式（函数）
6. 基表中的非空列没有包含在视图定义中
7. 包含集合操作符（UNION、UNION ALL、MINUS、INTERSECT）
8. 包含CONNECT BY或START WITH关键字
9. 部分连接查询

**6.6 修改视图定义**

1）需采用CREATE OR REPLACE VIEW语句修改视图的定义，实质其实是先删除后重建。需要重建、删除视图权限。

2）视图重定义后，依赖于该视图的其他视图以及PL/SQL程序都将失效（INVALID）。

**6.7 删除视图**

DROP VIEW XXX;

**6.8 查询视图信息**

|  |  |
| --- | --- |
| **视图名称** | **说明** |
| DBA\_VIEWS | 包含数据库中所有视图信息 |
| ALL\_VIEWS | 包含当前用户可以访问的所有视图信息 |
| USER\_VIEWS | 包含当前用户拥有的所有视图信息 |
| DBA\_UPDATABLE\_COLUMNS | 包含数据库所有视图中可更新的列信息 |
| ALL\_UPDATABLE\_COLUMNS | 包含当前用户可以访问的所有视图可更新的列信息 |
| USER\_UPDATABLE\_COLUMNS | 包含当前用户拥有的所有视图中的可更新的列信息 |

**6.9 实体化视图的刷新方法**

1）完全刷新（COMPLETE REFRESH）

重新根据实体化视图的定义，重新查询基表来生成数据。使用BULID IMMEDIATE子句创建视图。适合数据基本不变的情况。

2）快速刷新（FAST REFRESH）

利用实体化视图日志来更新实体化视图，又称增量刷新。采用快速刷新模式，需为实体化视图的每个基表建立实体化视图日志，保存基表中记录变更的日志信息。

**6.10 实体化视图刷新模式**

指定实体化视图的刷新时间，以保证实体化视图数据与基表数据保持一致，包括：

1. 需要时刷新（ON DEMAND）

仅在需要被刷新时，才进行刷新，可以手动通过DBMS\_MVIEW.REFRESH方法刷新视图，可以通过作业调度周期性刷新。

1. 提交时刷新（ON COMMIT）

一旦对基表执行了COMMIT操作，即事务提交，则立即刷新。会增加事务提交时间。

**6.11 创建实体化视图所需权限**

用户必须直接获得相应的权限，而不是通过角色获得该权限。

1. 在自己模式下所需权限

1、CREATE MATERIALIZED VIEW

2、CREATE TABLE

3、实体化视图用到的非当前用户下的基表的SELECT权限

1. 在其他模式下所需权限

1、CREATE ANY MATERIALIZED VIEW

2、CREATE ANY TABLE

3、实体化视图用到的非当前用户下的基表的SELECT ANY权限

1. 创建提交时刷新实体化视图所需额外权限

1、实体化视图所有基表的ON COMMIT REFRESH权限

**6.12 创建实体化SQL**

**CREATE MATERIALIZED VIEW 视图名 [列名1[,列名2]] [BUILD IMMEDIATE|DEFERRED] [WITH PRIMARY KEY | WITH ROWID] [ [REFRESH FAST|COMPLETE|FORCE] [ON DEMAND| COMMIT] [START WITH DATE] [NEXT DATE] ] | [NEVER REFRESH]**

**注：**

BUILD IMMEDIATE：创建视图是生成数据，默认设置

BUILD DEFERRED：创建视图视图不生成数据

WITH PRIMARY KEY：用于增量刷新中，视图日志通过主键的更改来刷新视图

WITH ROWID：用于增量刷新中，视图通过ROWID的更改来修改视图

RERRESH FAST：采用快速刷新模式

REFRESH COMPLETE：采用完全刷新模式

REFRESH FORCE：如果快速刷新可用，采取快速刷新，否则采取完全刷新。默认设置

ON DEMAND：需要时手动刷新

ON COMMIT：当基表发生事务提交时刷新

START WITH DATE：指定第一次刷新的时间

NEXT DATE：指定以后刷新的时间间隔

NEVER REFRESH：实体化视图不刷新

**6.13 创建手动刷新实体化视图**

CREATE MATERIALIZED VIEW TESTM AS SELECT \* FROM STUDENT;

**6.14 创建提交时刷新实体化视图**

CREATE MATERIALIZED VIEW TESTM ON COMMIT AS SELECT \* FROM STUDENT;

**6.15 创建增量刷新实体化视图**

1）创建视图日志

CREATE MATERIALIZED VIEW LOG ON 基表名 WITH ROWID;--如果有主键则通过主键刷新 WITH PRIMARY KEY

2）创建视图

CREATE MATERIALIZED VIEW TESTM REFRESH FAST ON COMMIT AS SELECT \* FROM STUDENT;

**6.16 创建定时刷新实体化视图**

CREATE MATERIALIZED VIEW TESTM START WITH SYSDATE NEXT SYSDATE + 1/1440 AS SELECT \* FROM STUDENT;

**6.17 维护实体化视图**

ALTER MATERIALIZED VIEW 视图名 [可用参数和创建时一样];

**6.18 删除实体化视图**

DROP MATERIALIZED VIEW 视图名;

**6.19 查询视图信息**

DBA\_MVIEWS、ALL\_MVIEWS、USER\_MVIEWS

1. **用户与权限管理**

**7.1 数据库预定义用户**

1）SYS：具有最高权限的数据库管理员，可以启动、修改和关闭数据库，拥有数据字典。

2）SYSTEM：辅助的数据库管理员，只是不能启动和关闭数据库。

3）SYSMAN：是OEM管理员

4）DBSNMP：是OEM代理，用来监视和管理数据库

**7.2 创建用户**

CREATE UERR 用户名 IDENTIFIED BY 密码 [DROP TABLESPACE 默认表空间名] [TEMPORARY TABLESAPCE 临时表空间名] [ACCOUNT LOCK|UNLOCK];

**7.3 修改用户**

ALTER USER [参数和创建用户一样];

**7.4 删除用户**

DROP USER 用户名 [CASCADE];

**7.5 查询用户**

|  |  |
| --- | --- |
| **视图名称** | **说明** |
| ALL\_USERS | 包含数据库所有用户的用户名、用户ID和用户创建时间等信息 |
| DBA\_USERS | 包含数据库说有用户的详细信息 |
| USER\_USERS | 包含当前用户的详细信息 |
| DBA\_TS\_QUOTAS | 包含所有用户的表空间配额信息 |
| USER\_TS\_QUOTAS | 包含当前用户的表空间配额信息 |
| V$SESSION | 包含用户会话信息 |
| V$SESSTAT | 包含用户会话统计信息 |

**7.6 权限概述**

ORACLE使用权限来控制用户对数据库的访问和用户在数据库中所能执行的操作。用户权限分为：

1. 系统权限：指在数据库级别执行某种操作的权限。如CREATE SESSION、CREATE ANY TABLE。
2. 对象权限：指针对某一特定对象的操作权限。如对表的增删改查权限

**7.7 系统权限授予**

ORACLE可以采用两种方式对用户进行授权：

1. 利用GRANT命令直接为用户授权

GRANT [权限名1[,权限名2..]] TO 用户名|角色名 [WITH ADMIN OPTION];

注：WITH ADMIN OPTION 表示用户可以将接收到的权限授予其他用户

例：GRANT CREATE SESSION TO SCOTT WITH ADMIN OPTION;

1. 先将权限授予角色，然后再将角色授予用户

GRANT [角色1[,角色2..]] TO 用户名;--授予角色不能用WITH ADMIN OPTION

例：GRANT CONNECT TO SCOTT;--CONNECT角色只包含CREATE SESSION权限

**7.8 系统权限回收**

REVOKE 权限名 FROM 用户|角色;

例：REVOKE CREATE SESSION FROM SCOTT;

**7.9 对象权限授予**

GRANT 权限名 ON 对象名 TO 用户名|角色 [WITH GRANT OPTION];

注：WITH GRANT OPTION 表示用户可以将接收到的权限授予其他用户

例：GRANT UPDATE ON POSIMM TO CLIENT;

**7.10 对象权限回收**

REVOKE 权限名 ON 对象名FROM 用户|角色 [CASCADE CONSTRAINTS]|[FORCE];

注：

CASCADE CONSTRAINTS：当回收REFERENCES对象权限或回收ALL PRIVILEGES对象权限时，删除利用RERERENCES对象权限创建的外键约束。

FORCE：当回收再表中被使用的用户自定义对象类型的EXECUTE权限时，必须知道FORCE关键字。

例：

REVOKE SELECT,UPDATE ON SCOTT.EMP FROM POSIMM;

**7.11 查询权限信息**

|  |  |
| --- | --- |
| **视图名称** | **说明** |
| DBA\_COL\_PRIVS | 包含数据库所有列对象的权限信息 |
| ALL\_COL\_PRIVS | 包含当前用户或PUBLIC用户拥有的或授予其他用户列对象的权限信息 |
| USER\_COL\_PRIVS | 包含当前用户拥有的或授予其他用户的所有列对象的权限信息 |
| DBA\_TAB\_PRIVS | 包含当前数据库中所有对象权限信息 |
| ALL\_TAB\_PRIVS | 包含当前用户或PUBLIC用户组拥有的所有对象的权限信息 |
| USER\_TAB\_PRIVS | 包含当前用户拥有的所有对象的权限信息 |
| DBA\_SYS\_PRIVS | 包含所有用户和角色获得的系统权限信息 |
| USER\_SYS\_PRIVS | 包含当前用户获得的系统权限信息 |
| SESSION\_PRIVS | 包含当前会话可以使用的所有权限信息 |

**7.12 创建角色**

CREATE ROLE 角色名 ;

**7.13 角色的授予、回收和删除**

1）授予用户角色

GRANT 角色名 TO 用户名;

2）回收用户角色

REVOKE 角色名 FROM 用户名;

3）删除角色

DROP ROLE 用户名;

**7.14 查询角色信息**

|  |  |
| --- | --- |
| **视图名称** | **说明** |
| DBA\_ROLES | 包含数据库中所有角色及其描述信息 |
| DBA\_ROLE\_PRIVS | 包含数据库中所有用户和角色授予的角色信息 |
| USER\_ROLE\_PRIVS | 包含当前用户授予的所有角色的信息 |
| ROLE\_ROLE\_PRIVS | 包含角色授予的角色信息 |
| ROLE\_SYS\_PRIVS | 包含角色授予的系统权限信息 |
| ROLE\_TAB\_PRIVS | 包含角色授予的对象权限信息 |
| SESSION\_ROLES | 包含当前会话具有的角色信息 |

1. **常用SQL**

**8.1 查看执行计划**

EXPLAIN PLAN FOR SELECT /\*+INDEX(T,POSHISIDX)\*/ COUNT(\*) FROM POSHIS T WHERE CN='CN20132809441' AND GPSTM>TO\_TIMESTAMP('2017-07-01 14:0:0','YYYY-MM-DD HH24:MI:SS') AND GPSTM<TO\_TIMESTAMP('2017-08-31 16:0:0','YYYY-MM-DD HH24:MI:SS');

SELECT \* FROM TABLE(DBMS\_XPLAN.DISPLAY);

**8.2 当前正在运行的SQL**

SELECT SID,

V$SESSION.USERNAME 用户名,

LAST\_CALL\_ET 持续时间,

STATUS 状态,

LOCKWAIT 等待锁,

MACHINE 用户电脑名,

TO\_TIMESTAMP(LOGON\_TIME) 开始登入时间,

SQL\_TEXT

FROM V$SESSION, V$PROCESS, V$SQLAREA

WHERE PADDR = ADDR

AND SQL\_HASH\_VALUE = HASH\_VALUE

AND STATUS = 'ACTIVE'

AND V$SESSION.USERNAME IS NOT NULL

ORDER BY LAST\_CALL\_ET DESC;

**8.3 查看锁情况**

SELECT SESS.SID,

SESS.SERIAL#,

LO.ORACLE\_USERNAME,

LO.OS\_USER\_NAME,

AO.OBJECT\_NAME,

LO.LOCKED\_MODE

FROM V$LOCKED\_OBJECT LO,

DBA\_OBJECTS AO,

V$SESSION SESS

WHERE AO.OBJECT\_ID = LO.OBJECT\_ID AND LO.SESSION\_ID = SESS.SID;

**8.4 解析时间最长的SQL**

SELECT \* FROM

(

SELECT

ROUND(AVG\_HARD\_PARSE\_TIME / 1000, 2)

AVG\_HARD\_PARSE\_TIME\_MS,

SQL\_ID,

EXECUTIONS

FROM

V$SQLSTATS

ORDER BY AVG\_HARD\_PARSE\_TIME DESC

)

WHERE

ROWNUM <= 10;

SELECT SQL\_TEXT FROM V$SQLAREA WHERE SQL\_ID = '14WNF35DAHB7V';

**8.5 查看磁盘读写最多SQL**

SELECT \* FROM (SELECT SQL\_TEXT,DISK\_READS "TOTAL DISK" , EXECUTIONS "TOTAL EXEC",DISK\_READS/EXECUTIONS "DISK/EXEC" FROM V$SQL WHERE EXECUTIONS>0 AND IS\_OBSOLETE='N' ORDER BY 4 DESC) WHERE ROWNUM<11;

**8.6 查看排序最多SQL**

SELECT HASH\_VALUE, SQL\_TEXT, SORTS, EXECUTIONS FROM V$SQLAREA ORDER BY SORTS DESC;

**8.7 磁盘争用**

SELECT NAME,PHYRDS,PHYWRTS,READTIM,WRITETIM FROM V$FILESTAT A,V$DATAFILE B WHERE A.FILE# = B.FILE# ORDER BY READTIM DESC;

**8.8 日志切换间隔**

SELECT B.RECID,TO\_CHAR(B.FIRST\_TIME,'DD-MON-YY HH:MI:SS') START\_TIME, A.RECID, TO\_CHAR(A.FIRST\_TIME,'DD-MON-YY HH:MI:SS') END\_TIME, ROUND(((A.FIRST\_TIME-B.FIRST\_TIME)\*25)\*60,2) MINUTES FROM V$LOG\_HISTORY A, V$LOG\_HISTORY B WHERE A.RECID = B.RECID+1 ORDER BY A.FIRST\_TIME DESC;

**8.9 连接数**

SELECT B.MACHINE, B.PROGRAM, COUNT (\*)

FROM V$PROCESS A, V$SESSION B

WHERE A.ADDR = B.PADDR AND B.USERNAME IS NOT NULL

GROUP BY B.MACHINE, B.PROGRAM

ORDER BY COUNT (\*) DESC;

**8.10 创建JOB**

1）SELECT \* FROM DBA\_JOBS;--查看JOB

2）EXEC DBMS\_JOB.BROKEN(144,TRUE);--停止JOB

3）EXEC DBMS\_JOB.REMOVE(144);--删除JOB

4）创建JOB

1、先创建一个存储过程

CREATE OR REPLACE PROCEDURE UPDATE\_SHUJUBAO

AS

BEGIN

UPDATE T\_SHUJUBAO SET YEWUCOUNT = (SELECT COUNT(\*) FROM T\_YEWU\_FLASH),YUNWEICOUNT = (SELECT COUNT(\*) FROM T\_YUNWEI\_FLASH),YEWUTM = (SELECT MAX(RCVTM) FROM T\_YEWU\_FLASH),YUNWEITM = (SELECT MAX(RCVTM) FROM T\_YUNWEI\_FLASH);

END;

2、通过定时任务，定时执行存储过程

VARIABLE JOBB NUMBER;

BEGIN

DBMS\_JOB.SUBMIT(:JOBB,'UPDATE\_SHUJUBAO;', SYSDATE, 'SYSDATE+1/1440');--关键是UPATE\_SHUJUBAO后的冒号

END;

**附录：**

1. 用户账号状态描述（ACOUNT\_STATUS）
2. SELECT \* FROM USER\_ASTATUS\_MAP;--状态几种类型
3. SELECT USERNAME,ACCOUNT\_STATUS,DEFAULT\_TABLESPACE,TEMPORARY\_TABLESPACE FROM DBA\_USERS;--所有的用户信息
4. SELECT \* FROM ***[DBA\_PROFILES](#DBA_PROFILES)***;--数据库的基本信息
5. SELECT \* FROM DBA\_PROFILES WHERE RESOURCE\_NAME='FAILED\_LOGIN\_ATTEMPTS';--输入密码不对，锁定账号的尝试次数
6. ALTER PROFILE DEFAULT LIMIT FAILED\_LOGIN\_ATTEMPTS 20;--更改锁定尝试次数，UNLIMITED为无限
7. SELECT \* FROM DBA\_PROFILES WHERE RESOURCE\_NAME='PASSWORD\_LOCK\_TIME';--密码输入次数锁定后，设置自动解锁时间
8. ALTER PROFILE DEFAULT LIMIT PASSWORD\_LOCK\_TIME 1/24;--修改自动解锁时间为2小时
9. 同理EXPIRED(GRACE) RESOURCE\_NAME = ‘PASSWORD\_GRACE\_TIME’;

|  |  |  |
| --- | --- | --- |
|  | **状态** | **描述** |
| 0 | OPEN | 当前帐户是开放的用户可以自由登录 |
| 1 | EXPIRED | 当前帐户已经过期, 用户必须在修改密码以后才可以登录系统, 在登录的时候, 系统会提示修改密码 |
| 2 | EXPIRED(GRACE) | 这是有password\_grace\_time定义的一个时间段, 在用户密码过期以后的第一次登录, 系统会提示用户,密码在指定的时间段以后会过期, 需要及时修改系统密码. |
| 4 | LOCKED(TIMED) | 这是一个有条件的帐户锁定日期, 由password\_lock\_time进行控制, 在lock\_date加上password\_lock\_time的日期以后,帐户会自动解锁. |
| 8 | LOCKED | 帐户是锁定的,用户不可以登录, 必须由安全管理员将帐户打开用户才可以登录. |
| 5 | EXPIRED & LOCKED(TIMED) |  |
| 6 | EXPIRED(GRACE) & LOCKED(TIMED) |  |
| 9 | EXPIRED & LOCKED |  |
| 10 | EXPIRED(GRACE) & LOCKED |  |

1. 用户信息描述（DBA\_USERS）
2. SELECT \* FROM DBA\_USERS;--所有信息
3. 一般用来获取表空间信息

|  |  |  |  |
| --- | --- | --- | --- |
| **Column** | **Description** | **Column** | **Description** |
| USERNAME | 用户名 | CREATED | 用户创建日期 |
| USER\_ID | 用户ID | PROFILE | DEFAULT |
| PASSWORD | 用户密码 | EXTERNAL\_NAME | NULL |
| LOCK\_DATE | 账号锁定时的日期 | PASSWORD\_VERSIONS | 10G 11G |
| EXPIRY\_DATE | 账号失效日期 | EDITIONS\_ENABLED | 保留字 |
| DEFAULT\_TABLESPACE | 默认表空间 | ACCOUNT\_STATUS | 用户账号状态 |
| TEMPORARY\_TABLESPCE | 默认临时表空间 |  |  |

1. 数据库信息描述（DATABASE\_PERPERTY）

1）SELECT \* FROM DATABASE\_PROPERTIES;--所有信息

|  |  |  |
| --- | --- | --- |
| **PerpertyName** | **PerpertyValue** | **Description** |
| DEFAULT\_TEMP\_TABLESPACE | TEMP | 数据库默认临时表空间 |
| DEFAULT\_PERMANENT\_TABLESPACE | USERS | 数据库默认表空间 |
| DEFAULT\_TBS\_TYPE | SMALLFILE | Default tablespace type |
| NLS\_LANGUAGE | AMERICAN | Language |
| NLS\_CHARACTERSET | ZHS16GBK | Character set |
| NLS\_CALENDAR | GREGORIAN | Calendar system |
| NLS\_DATE\_FORMAT | DD-MON-RR | Date format |
| NLS\_DATE\_LANGUAGE | AMERICAN | Date language |
| NLS\_SORT | BINARY | 排序算法 |
| NLS\_TIME\_FORMAT | HH.MI.SSXFF AM | Time format |
| NLS\_TIMESTAMP\_FORMAT | DD-MON-RR HH.MI.SSXFF AM | Time stamp format |
| NLS\_NCHAR\_CHARACTERSET | AL16UTF16 | NCHAR Character set |
| NLS\_RDBMS\_VERSION | 11.2.0.1.0 |  |
| GLOBAL\_DB\_NAME | VITS | Global database name |
| NO\_USERID\_VERIFIER\_SALT | 55A6B9F2E250FA25D976C941F284F640 |  |

1. DBA\_PROFILES

|  |  |  |  |
| --- | --- | --- | --- |
| COMPOSITE\_LIMIT | LOGICAL\_READS\_PER\_SESSION | PRIVATE\_SGA | PASSWORD\_REUSE\_MAX |
| SESSIONS\_PER\_USER | LOGICAL\_READS\_PER\_CALL | IDLE\_TIME | **PASSWORD\_LOCK\_TIME** |
| CPU\_PER\_SESSION | **FAILED\_LOGIN\_ATTEMPTS** | **CONNECT\_TIME** | **PASSWORD\_GRACE\_TIME** |
| CPU\_PER\_CALL | **PASSWORD\_LIFE\_TIME** | PASSWORD\_REUSE\_TIME | PASSWORD\_VERIFY\_FUNCTION |

1. 临时表空间数据文件信息（DBA\_TEMP\_FILES）
2. SELECT \* FROM DBA\_TEMP\_FILES;

|  |  |  |  |
| --- | --- | --- | --- |
| **Column** | **Description** | **Column** | **Description** |
| FILE\_NAME | 文件名 | RELATIVE\_FNO | 相对文件号 |
| FILE\_ID | 文件ID | AUTOEXTENSIBLE | 是否自动扩展 |
| TABLESPACE\_NAME | 表空间名 | MAXBYTES | 文件最大大小 |
| BYTES | 文件大小 | MAXBLOCKS | 文件最大块数 |
| BLOCKS | 文件块数 | INCREMENT\_BY | 默认的自动扩展大小 |
| STATUS | 文件状态 | USER\_BYTES | 已用大小 |
| USER\_BLOCKS | 已用块数 |  |  |

1. 回退段状态（UNDO\_STATUS）

|  |  |
| --- | --- |
| **状态** | **描述** |
| ACTIVE | 回退信息对应事务没有被提交 |
| UNEXPIRED | 回退信息对应事务已经提交，但保留时间没有超过保留期 |
| EXPIRED | 回退信息对应事务已经提交，保留时间超过保留期 |

1. 查询表空间信息

|  |  |
| --- | --- |
| **Column** | **Description** |
| DBA\_TABLESPACES | 包含数据库所有表空间信息，包含临时表空间 |
| USRE\_TABLESPACES | 包含数据库中当前用户可以使用的表空间信息 |
| DBA\_DATA\_FILES | 包含所有永久数据文件及其所属表空间 |
| DBA\_TEMP\_FILES | 包含所有临时数据文件及其所属临时表空间信息 |
| DBA\_USERS | 包含所有用户的信息，比如用户的默认表空间和临时表空间 |
| DBA\_FREE\_SPACE | 表空间中空闲区的信息 |
| USER\_FREE\_SPACE | 当前用户 |
| V$DATAFILE | 同DBA\_DATA\_FILES比其好，需与V$TABLESPACE配合，查询表空间名 |
| V$TEMPFILE | 同DBA\_TEMP\_FILES比其好，同上 |
| V$TABLESPACES | 同DBA\_TABLESPACE比其好 |
| DBA\_SEGMENTS | 表空间中的段信息，数据段，索引段，回退段，临时段 |
| USER\_SEGMENTS | 当前用户，用来查某个表空间包含哪些表和索引 |
| DBA\_TEMP\_FREE\_S |  |
| DBA\_UNDO\_EXTENT | 因为临时表空间只有临时段，所以没有DBA\_TEMP\_SEGMENTS选项 |

1. 数据文件、表空间、段、区、块

|  |  |
| --- | --- |
| **Column** | **Description** |
| 数据文件（DATAFILE） | 数据库存储数据的物理单元，一个数据文件只属于一个表空间 |
| 表空间（TABLESPACE） | 数据库逻辑上有多个表空间组成，一个表空间可以包含多个数据文件 |
| 段（SEGMENT） | 表空间中储存的对象，一个表空间包含0个或多个段 |
| 区（EXTENT） | 段由区组成，区是磁盘分配的最小单位。段的增大是通过增加区的个数来实现的。每个区的大小是数据块大小的整数倍，区的大小可以不相同 |
| 块（BLOCK） | 数据块是数据库中最小的I/O单位，同时也是内存数据缓冲区的单位，及数据文件存储空间单位。块的大小由参数DB\_BLOCK\_SIZE设置，其值应设置为操作系统块大小的整数倍。 |