一、oracle体系架构

* 闪回 flashback ：

误删表，解决逻辑错误。

oracle instance 例程

user\_自己的

all\_能访问的

dba\_系统的，有权限才能访问的

select name from v$datafile;

* 更新统计数据

analyze table emp compute statistics

restore：backup备份，从备份文件中还原

recover：恢复，前滚，通过表空间

create tablespace demo datafile '/u01/app/oracle/oradata/db12c/demo01.dbf' size 10m;

create table scott.st（id int, name varchar(20)) tablespace demo;

insert into scott.st values(1,'Tom');

commit;（发给LGWR）

select \* from scott.st;

desc dba\_data\_files;

select FILE\_NAME from dba\_data\_files where tablespace\_name=(select tablespace\_name from dba\_tables where table\_name='ST');

返回oracle: rm demo01.dbf

返回数据库: alter database create datafile 5;

recover datafile 5;

alter database open;

数据库识别逻辑，操作系统识别物理。

系统数据用户数据不要放在一起，性能会下降。

conn scott/123456@db12c 通过服务名解析

添加服务：netmgr

user跟server之间还要有监听器：沟通用户和服务器进程

监听器配地址跟数据库。

lsnrctl status

lsnrctl off 关闭监听

lsnrctl start 开启监听，动态的，有延迟，需要花费时间寻找服务

静态注册，无延迟：用netmgr添加监听程序->添加数据库

配了静态后动态更慢

在admin目录下：cat listener.ora

* 后台进程：

1. pmon 进程监视器进程

用来监视用户进程，出现异常即处理，

2. smon 系统监视器进程

用来监视系统，改完数据系统重启了，还没来得及写入文件，->写入日志，例程恢复。

例程恢复：内存中产生的还未来得及写入文件。

介质恢复：已写入文件，文件坏了。

3. dbwr 数据库书写器进程

eg：dbw0，dbw1，可配多个

数据库高速缓存的数据写入数据文件（内存写入外存）。

4. lgwr 日志书写器进程

重做日志缓存中的数据写入重做日志文件。

5. ckpt 检查点进程

checkpoint

eg：当前数据库与一分钟前的不一样：

scn：system change number 系统改变号，在不断变化，是和时间相关的单调递增序列。

即恢复时将数据库推到某个点（scn）上。

checkpoint\_change\_number：检查点号，相对静止。

scn>=检查点

检查点作用：

alter system checkpoint;做检查点

1）同步所有的数据文件（每个数据文件头部都有检查点号）

select name checkpoint\_change# from v$datafile;

2）同步所有的控制文件

3）发送信号通知DBWR写盘（发送上一次检查点之后的数据）

检查点间隔时间：根据业务需求设计

短：减少例程恢复时间，检查点为io密集型时间，性能下降

长：性能好，时间长。

show parameter mttr：

fast\_start\_mttr\_target：设置检查点多久一次，0为禁用功能

alter system set fast\_start\_mttr\_target=300；出了问题恢复需要300s。

现在数据库出问题需要多久恢复？

不知道，desc v$instance\_recovery

select ESTIMATED\_MTTR from v$instance\_recovery；查询

ESTIMATED\_MTTR累计到300s触发一次检查点。

查询后台进程：

desc v$bgprocess

select NAME from v$bgprocess;

* 服务器进程：

输入命令，查library cache中是否有，有直接用

数据：database buffer cache。

* alter system与alter database：

如果操作对instance，即system；对数据库操作即database。

二、数据库

1. oracle内部文件（灰色）

由oracle处理，select name from v$controlfile；都带scn号

data file：数据文件。

control file：控制文件。

redo log file：select member from v$logfile，看日志文件。

2.oracle外部文件（蓝色）

由dba（管理员）管理，只要不改永远不变，show parameter spfile，性质等同于操作系统文件。

parameter file：初始化参数文件，初始化配置。

password file：口令验证文件：随便输conn sahojdah/shioa as sysdba都可以通过，操作系统验证，修改。

archived log file：归档日志文件。

3.oracle日志工作体系

数据库日志模式：

1）非归档模式：日志不保留 直接覆盖

2）归档模式：覆盖前产生归档日志文件

三、SQL语句

1.QL查询语句

select

与表无关查dual 系统表：select 3+4 from dual；

2.DML增删改语句

insert update delete merge

3.DDL定义

create alter drop truncate rename comment，修改系统数据

4.DCL数据控制语句

grant给权限，revoke取消权限

grant select on hr.departments to scott;

revoke from；

5.TCL事务控制语句

commit rollback savepoint

help index 出现所有sqlplus命令，不产生歧义都可以缩写

help @，@：调用脚本

* 考试：实现TOP n功能

使用行限制子句，

1.fetch first

select empno,ename sal from scott.emp order by sal desc fetch first 2 rows only

2.with ties 有数据一样的

select empno,ename sal from scott.emp order by sal desc fetch first 2 rows with ties（出现三行）

3.offset first n rows偏离

select empno,ename sal from scott.emp order by sal desc offset 3 rows fetch first 2 rows only（第四五行）

* 和用户交互

利用提交变量

1. & 查询

接收用户的输入

select \* from scott.emp where empno=&工号 ，即可输入工号查询

select \* from scott.emp where ename=UPPER('&姓名');

2. define

define gh=7369

select \* from scott.emp where empno=&gh，不用输，直接出来了

undefine gh取消定义

select empno,ename,&c3 from scott.emp order by &c3;要输两次c3

3.&&

select empno,ename,&&c3 from scott.emp order by &c3；输一次

接收用户输入，定义

* sqlplus配置

1.show 查看

show all 所有

show time 查看时间

2.set 设置

set time on 时间显示打开

set time off 时间显示关闭

set head/feed on/off 去头尾

col sal noprint 不显示工资列

以上只有当前生效，永久要进入admin的文件中改

* 函数

注意含义、参数重载、应用。

查询oracle文档docs.oracle.com->oracle database->12.2版本->get start->development->SQL Language Reference

1. 分类

单行函数，每一行都返回一个值（职员有哪些）。多行函数/组函数，对多条处理，然后返回值（平均工资）。

2.举例

select CONCAT（‘Hello’，‘World’） from dual;->HelloWorld

select SUBSTR（‘HelloWorld’,1,5） from dual;->Hello 取子串

select SUBSTR（‘HelloWorld’,-5,5） from dual;->World 从右边开始数取子串

select LENGTH（‘HelloWorld’） from dual;->6

select INSTR（‘HelloWorld’,‘W’） from dual;->6 W在第六位

select INSTR（‘HelloWorld’,‘l’，4） from dual;->4 从第4位开始找

select INSTR（‘HelloWorld’,‘l’，1,2） from dual;->从第1位找第二个出现的

LPAD(last\_name,12,'-')找12位的last\_name，不足的位数左填充-：---------Tom

可用来格式化层次化数据，层次查询：SCOTT表

层次查询需要定义起点和方向

select empno,ename,mgr from emp;（推导领导的领导）

oracle提供了伪列level

select lpad(' ',(level-1)\*2)||ename ename,level from emp start with empno=7839 connect by prior empno=mgr;(找下属）

RPAD右填充

select TRIM（‘H’from‘HelloWorld’） from dual;->elloWorld 截断

ROUND(45.926,2)->45.93 四舍五入到小数点后2位

ROUND(45.926,-1)->50 四舍五入到个位数

TRUNC(45.926,2)->45.92 向下取整到小数点后2位

MOD（1600，300）->100 取余数

CEIL（2.83）->3 向上取整

FLOOR（2.83）->向下取整

3.日期相关

select sysdate from dual;返回系统日期，sysdate current\_date

nls参数：n国家 l语言 s支持

日期格式参数：nls\_date\_format

alter session set nls\_date\_format='yyyy-mm-dd hh:mi:ss';修改查询日期格式（设置在会话级生效）

alter session set nls\_date\_format='yyyy-mm-dd hh24:mi:ss'; 改为24小时制

alter session set nls\_date\_format='yyyy-mm-dd hh:mi:ss am';用am pm表示

systimestamp,current\_timestamp 时区

select systimestamp（3） from dual;返回3位

日期可以运算：date相减变为number。

千年虫问题：

用两位数表示年份

1）YY格式

指定目标日期时要用系统日期判断，和当前日期处于同一个世纪。

2）RR格式

数据库中的默认值，返回接近于当前年代的世纪

MONTHS\_BETWEEN('01-SEP-05','11-JAN-04') 两个日期间相差多少个月，离当前近的日期写前面，离当前远的日期写后面

ADD\_MONTHS('01-SEP-05',1)该月添加一个月

NEXT\_DAY('01-SEP-05','FRIDAY')；NEXT\_DAY('01-SEP-05',6)该日后的第一个星期五

ROUND（SYSDATE,'MONTH'）;舍入 SYSDATE=‘25,-JUL-95’ 01-AUG-95

TRUNC（SYSDATE,'MONTH'）; SYSDATE=‘25,-JUL-95’ 01-AUG-95

4.转换函数

to开头的

TO\_CHAR，TO\_NUMBER，TO\_DATE

显式转换；隐式转换：不用函数，自动能转。

TO\_CHAR(date[,'format\_model'])用有效的格式模型，fm移除空格

select ename,to\_char(hiredate,'yyyy-mm-dd') from emp;

select ename,to\_char(hiredate,'yyyy-month-dd') from emp;将月份拼写出来

select to\_char(sal,'99,999.99') from emp; 工资：1,800.00

查参数 nls\_sessions\_parameters,nls\_instance\_parameters

select \* from nls\_sessions\_parameters；

update emp set HIREDATE=to\_date('1981-09-05','yyyy-mm-dd') where empno=7900;字符串转换成日期处理

5.处理null的函数

1)NVL(expr1,expr2)

空显示expr2的值，不为空显示expr1的值

select empno,ename,comm,nvl(comm,0) from emp;

2)NVL(expr1,expr2)

空显示expr3的值，不为空显示expr2的值，空值参与运算的问题

select empno,ename,sal,comm,nvl2(comm,comm+sal,sal) from emp;

3)NULLIF(expr1,expr2)

expr1与2值相同为空，不一样显示expr1

4）COALESCE(expr1,expr2,...,exprn)

找到第一个非空值为止

6.处理if-then-else逻辑

1）CASE语句

CASE expr WHEN comparison\_expr1 THEN return\_expr1

WHEN comparison\_expr2 THEN return\_expr2

.....

ELSE exprn

END

2）DECODE函数

DECODE(job\_id,'IT\_PROG',result1,

'ST\_CLERK',result2,

salary)

确定的值用decode方便，带范围的只能用case

7.组函数

AVG MIN MAX COUNT SUM

greatest（1,3,5）->5;最大的数

LISTAGG 排序

STDDEV 偏差

VARIANCE 方差

select LISTAGG(ename,',') WITHIN GROUP (ORDER BY deptno) ename from emp GROUP BY septno;

GROUP BY 分组

HAVING WHERE区别：

WHERE记录筛选

HAVING分组筛选

select department\_id,job\_id,sum(salary) from employees GROUP BY rollup(department\_id,job\_id) ORDER BY department\_id;根据部门号小计

rollup单向小计

cube两个方向都小计

select department\_id,avg(salary) from employees GROUP BY department\_id HAVING avg(salary)>8000;

8. 导入数据文本中数据

sqlldr加载数据

load

infile '/u01/text.dat'

into table hr.st

(name char terminated by ',',

subject char terminated by ',',

score integer external terminated by ',')

sqlldr hr/hr control=text.ctl

* 访问多张表

1.连接

1）内部连接

select b.buyer\_id,b.buyer\_name,s.qty from buyers b,sales s where b.buyer\_id=s.buyer\_id;

select b.buyer\_id,b.buyer\_name,s.qty from buyers b,sales s where b.buyer\_id=s.buyer\_id and b.buyer\_id=1;

将能筛选掉最多纪录的条件写在最后面--->sql语句优化

eg：

dept（4个部门）

emp（10万员工）

from emp,dept---->小表放后面大表放前面

放在后面的表称为基准表/驱动表，nested loop嵌套循环两张表各访问四次

2）外部连接（左/右/全外连接）

——不匹配的纪录也要显示

左外连接：左边表全显示

select b.buyer\_id,b.buyer\_name,s.qty from buyers b,sales s where b.buyer\_id=s.buyer\_id(+);

select b.buyer\_id,b.buyer\_name,s.qty from buyers b left join sales s on b.buyer\_id=s.buyer\_id;

右外连接：

select b.buyer\_id,b.buyer\_name,s.qty from buyers b,sales s where b.buyer\_id(+)=s.buyer\_id;

select b.buyer\_id,b.buyer\_name,s.qty from buyers b right join sales s on b.buyer\_id=s.buyer\_id;

全外连接：

select b.buyer\_id,b.buyer\_name,s.qty from buyers b full join sales s on b.buyer\_id=s.buyer\_id;

3）多表连接（>=3张表）

=两个两两连接

select b.buyer\_name,p.prod\_name,s.qty from buyers b,sales s,product p where b.buyer\_id=s.buyer\_id and p.prod\_id=s.prod\_id;

select b.buyer\_name,p.prod\_name,s.qty from buyers b join sales s on b.buyer\_id=s.buyer\_id join product p on p.prod\_id=s.prod\_id;

四张表之后性能下降，使用实体化视图（物化视图）。

4）自连接

给一张表起两个不同的别名

select a.buyer\_id as buyer1,a.prod\_id,b.buyer\_id as buyer2 from sales a,sales b where a.prod\_id=b.prod\_id and a.buyer\_id<>b.buyer\_id;

5）交叉连接

一张表的所有纪录与另一张的所有纪录都要匹配。

select b.buyer\_name,s.qty from buyers b,sales s；

sql3方法：

自然连接：natrual join 用两张表中一样的字段来匹配，如果没有相同字段会做交叉连接，如果有两个字段匹配则两个都要满足。

using：指定匹配字段

2.子查询

1）嵌套子查询

2）关联子查询

单行操作符：=

多行操作符：in all any

select empno,ename sal from emp where sal in (1500,2850);

select empno,ename sal from emp where sal >any (1500,2850);

select empno,ename sal from emp where sal >all (1500,2850);

3. 合并结果集：

1）union all

合并

2）union

相同的部分留一份,去掉重复记录

select employee\_id,job\_id from employees union select employee\_id,job\_id from job\_history;

3）intersect

取交集

4）minus

结果集1减去结果集2重复的

* DML数据操纵语句（改用户数据）

1.insert into table[(column,...)] values(column,...);

值一定要给全，不能省略

2.update table set ... = ...；

3.delete from table where ....；

savepoint xxx;

之后删错了，rollback到xxx；

4.merge合并

merge into table using （ ） on ... when matched then update set ...=... when not matched then insert values（...,...）

merge into emp\_hz h using emp\_gz g on (h.empno=g.empno)

when matched then

update set h.SAL=g.SAL, h.ename=g.ename

when not matched then

insert values（g.empno,g.ename,g.sal）;

* TCL语句

1.commit

2.rollback

3.savepoint

DDL语句（alter包含自动commit）

DDL DCL语句都会自动提交

grant select on emp\_hz to hr; scott把访问emp\_hz的权限给hr。

* Read consistency

读一致性：事务开始前与开始后看到的是一样的。

修改之前，将数据块复制进回滚段，碰到select，尚未commit，该语句被锁住，select去访问回滚段。

去回滚段访问，称为读一致性。

回滚段的作用：考！！！

1.读一致性

读的人不用等写的 写的不用等读的

2.回滚

删错了后悔尚未commit，之前被删的在回滚段中，rollback即回来了。

3.闪回恢复

删错了且commit了，

select \* from emp\_hz as of timestamp(systimestamp-interval '2' minute);两分钟前数据还在

insert into emp\_hz select \* from emp\_hz as of timestamp(systimestamp-interval '2' minute);

commit；即恢复了

* 锁：保证串行化操作

1）T1：修改行h1，锁定h1

2）T2：修改行h2，成功，行h2也被锁定了，select \*，T1修改的看不到。

3）T1：T2修改的也看不到

4）T1要改h2，做不了，要等T2结束释放h2.

5）T2再改h1，死锁了，T1的update命令被回滚了。

但是有情况不会回滚，一直死锁。

解决方法：某客户端一直不commit，其他客户端一直在等

sqlplus / as sysdba

desc v$lock——BLOCK 阻塞=1，未阻塞=0

select sid from v$lock where block=1;——60

desc v$session 找SID=60的

select sid,serial# from v$session where sid=60;——serial#=34289

alter system kill session '60,34289';杀掉

* DDL语句

与DML不同：

增删改系统数据，DML为用户数据

1.create

起名规则：

新建表不能与同一个用户的相同名称空间下的对象同名，字母开头

select OBJECT\_NAME,OBJECT\_TYPE,NAMESPACE from user\_objects;查名称空间

表名不可以是保留关键字

建表规则：

有建表权限，表空间还剩余空间

create user demo1 identified by abc; 新建一个口令为abc的用户demo1

grant create session to demo1;

grant create table to demo1;还不能建表

create table t1(id int);建好了，但是无法插入数据->尚未分配表空间，且没有使用权

desc dba\_users 查用户默认表空间

select DEFAULT\_TABLESPACE from dba\_users where USER\_NAME='DEMO1'

show parameter defer 延时创建参数 value改为false 第一次使用时分配空间

alter system set ...= false

alter user demo1 quota 5m on users;让demo1用户可使用users表空间的5M空间，即有使用权限了

（alter user demo1 quota unlimited on users;）

字段规则：

起名不能同名，不能是伪列

伪列：存在、可查、但不依赖于表。

eg：rowid：数据在文件号+块号+行号中（物理地址：使用BASE64 CODE），可用dbms\_rowid解析

select empno,dbms\_rowid.ROWID\_RELATIVE\_FNO(rowid) File#,dbms\_rowid.ROWID\_BLOCK\_NUMBER(rowid) BLOCK#

dbms\_rowid.ROWID\_ROW\_NUMBER(rowid) ROW# from emp;

rownum行号，查询时产生结果集，每碰到一条记录，该记录编号，编号从1开始

nvarchar和varchar：

字符集/编码方案（eg：ASCII，unicode）

dbca数据库配置助手

nvarchar字符来自国家字符集，字符集选用Unicode，定长编码 性能影响

char/varchar来自数据库字符集

row类型：

存未加工的二进制位串（字符的二进制形式，便于机器看）

2.alter改

1）改表名

alter table t11 rename to t1; t11改名为t1

2）添加字段

alter table buyers add birthdate date; 给buyers表添加birthdate字段

3）字段改名

alter table buyers rename column birthdate to birth\_date;

4）修改字段类型

alter table buyers modify BIRTH\_DATE char(8);空字段可以直接改

5）修改字段宽度

alter table buyers modify BUYER\_NAME varchar(30);从varchar20改到varchar30，改大可以

char改为varchar不行，char为定长

6）删除字段

①直接删

alter table buyers drop column BIRTH\_DATE;

alter table buyers drop (BIRTH\_DATE);

②先标记为未使用然后删除

alter table buyers set unused column BUYER\_NAME;

alter table buyers drop unused column;

表比较小/表不小但空闲时间直接删，大表要先未使用，等到变为空闲之后再删。

3.drop删除

drop table buyers;

show recyclebin 回收站中还有，buyers未完全删掉

flashback table buyers to before drop;就弄回来了

drop table buyers purge; 回收站中也删掉了

4.truncate清空表中记录

truncate与delete区别：

①truncate属于DDL语句、快，delete属于DML语句、慢

delete删之后要先放到回滚段，可以rollback回来，truncate不往回滚段放，直接commit，无法rollback。

②truncate执行完会释放存储空间（block），delete不释放。

HWM（high water marker 高端水印/高水位标记）：

oracle扫描时扫描到HWM为止，计算剩余空间只计算HWM之上的

5.rename给表改名

rename t1 to t11;表t1改为t11

6.comment

给表加注释

comment on table buyers is 'This is a demo table.';

desc user\_tab\_comments

select COMMENTS from user\_tab\_comments where TABLE\_NAME='BUYERS';

给column加注释

comment on column product.sal is 'This is a demo column‘；

select COMMENTS from user\_col\_comments where TABLE\_NAME='PRODUCT';

* Oracle数据库常见对象

Table表，View视图（虚表），Sequence序列（可产生编号），Index索引，Synonym同义词（起别名 缩短名称）

create synonym emp for scott.emp;

* 数据的完整性

数据+加规则——>保证数据是有效的合理的

实现方法

1.代码

2.触发器

3.约束

能用约束就尽量用约束，开销小。

* 约束

1.约束类型

1)not null非空

2)unique不能重复

3)pk主键不允许空也不允许重复

4)fk外键（子表字段必须来自父表字段）

ON DELETE CASCADE 级联删除（删父表时连带删子表记录）

ON DELETE SET NULL 置空 不删

5)check

eg：限制工资大于0

2.查询约束信息

desc user\_constraints

desc user\_cons\_column看到约束加在哪个列的

约束要起名，如果没有起名（可能该约束不用去管理），系统起名SYS\_Cn

3.定义约束

1）列级

紧跟创建表时的列之后约束

constraint 约束名 约束类型

2）表级

最后约束，需要起名

constraint 约束名 约束类型(字段名)

4.删约束

ALTER TABLE xxx CONSTRAINT ...

5.约束禁用

当外部大批数据要进入表时，表中有约束，插入每一条都会检查约束->非常慢

插入后，插入的数据有些不满足约束

使用EXCEPTIONS表,使用urlexcpt.sql脚本

@？/rdbms/admin/utlexcpt

？代表oracle\_home

查oracle脚本：

cd $oracle\_home

cd rdbms/

cd admin

ls

alter TABLE test ENABLE CONSTRAINT uni\_name EXCEPTIONS INTO exceptions;会报错，是为了把违反约束的放入表中

做完清空exceptions表

* view视图

1.视图的作用

收集感兴趣的数据，屏蔽敏感数据，简化权限管理

2.视图的分类

1）简单视图：只有一张表 不包含分组数据或函数

2）复杂视图：>=一张表,....

3.定义视图

grant create view to scott;

create view emp\_info as select empno,ename,sal from emp;

4.修改视图

1）CREATE OR REPLACE view emp\_info as select empno,ename,sal,deptno from emp;

如果先drop再create，则为新对象，权限要重新赋

2）CREATE OR REPLACE FORCE view emp\_info as select empno,ename,sal,deptno from abc;

创建但有警告（编译错误），abc表不存在，当什么时候有abc表什么时候通过。

强制创建视图，就算表不存在

3）update emp\_info set sal=1800 where empnp=7369;

视图改成功了，表中也变了

4）CREATE OR REPLACE FORCE view emp\_info as select empno,ename,sal,deptno from abc where deptno=30;

当表中有数据deptno改为非30，即不会在视图中显示

5）CREATE OR REPLACE FORCE view emp\_info as select empno,ename,sal,deptno from abc where deptno=30 with check option;

deptno=30不能改，改会报错

6）CREATE OR REPLACE FORCE view emp\_info as select empno,ename,sal,deptno from abc where deptno=30 with read only;

只读视图，什么都不能改

想要改的话再 create or replace不带with read only一遍即可

7）规则

如果视图中包含组函数，group从句，distinct关键字，伪列，都不能修改

列由表达式（其他几列计算得到的结果）定义，不能做update、insert

创建视图时，有些列约束为非空，视图不能做insert操作。

创建表时，列的非空约束继承，主键不继承

5.内嵌视图

select xxx from (select xxx from table ...) where ...；对表进行预处理

select rownum,empno.ename from (select rownum aa,emp.\* from emp) where aa>=12;找最后三条数据

select rownum,empno.ename from emp where rownum>=12 一条都找不到

* 例程管理

1.关闭（四种模式）AITN

1）正常关闭（默认模式）NORMAL

shutdown=shutdown normal

不一定能正常关，要等所有会话关闭

2）事务型关闭TRANSACTIONAL

要等当前事务结束

3）立即关闭IMMEDIATE

不需要等事务之类的

一般都用I关闭，NTI都不会丢数据，因为会做检查点

4）终止退出ABORT

不做检查点，其他都做

2.启动（三个阶段）

select status from v$instance

1）例程启动（no mount阶段）STARTED

需要访问初始化参数文件，只能访问部分动态性能视图（内容来自内存）（不能访问v$datafile），内容来自控制文件访问不了

cd $ORACLE\_HOME

cd dbs

ls

$ spfiledb12c.ora（初始化参数文件）

2）加载数据库（mount阶段）

需要访问控制文件，可以访问所有的动态性能视图（所有的v$），数据文件访问不了

startup mount/alter database mount

初始化参数文件中指定了一个控制文件，控制文件中记录了整个数据库状态

show parameter control\_files

3）打开数据库（open阶段）

startup=startup open

需要访问联机重做日志文件和数据文件（redo01.log）

可以访问所有文件

* 配置

1.初始化参数文件

1)文本文件（pfile）

完全用可显示字符组成的，可用任何文本编辑器修改

init<sid>.ora ---initdb12c.ora

2)二进制文件（spfile）serverpfile

不是完全用可显示字符组成的，pfile每次改完都要重启才能生效，不能用文本编辑器改，要用命令改

改完立即写入服务器，立即生效

spfile<sid>.ora ---spfiledb12c.ora

2.查看初始化参数

show parameter

3.修改初始化参数

alter system set ...=...

静态参数不能直接改（false）需要先写到spfile 重启才能生效，动态参数直接改就能生效（immediate）

deferred延时参数，重连才能生效

scope子句：

memory修改完不写到初始化参数文件中

spfile

both

4.将初始化参数还原成默认值

alter system set shared\_servers=10;

alter system reset shared\_servers;

重启才能生效（shutdown immediate，startup nomount）

所有参数还原为默认值：初始化参数中所有参数都不标注具体数值

5.修复错误的初始化参数

（可查看警告日志文件，找到哪个参数出错）

根据spfile创建pfile，然后修改pfile

create pfile from spfile;

vim initdb12c.ora

修改错误的地方

mv spfiledb12c.ora spfiledb12c1.ora //不用spfile启动

stratup nomount //此时会用pfile启动

根据pfile创建spfile

6.例程启动时选择初始化参数文件的顺序

1）spfile<sid>.ora ---spfiledb12c.ora

2）spfile.ora

3）init<sid>.ora ---initdb12c.ora

7.利用指定的初始化参数文件启动

cp initdb12c.ora /u01/a.ora

startup pfile=/u01/a.ora

8.查看oracle管理手册看参数用法

oracle administrator's guide

oracle database reference

9.查看警告日志文件

show parameter dump

/u01/app/oracle/diag/rdbms/db12c/trace/alert\_db12c.log

只要改错了就一定会产生

* 手工创建数据库（sales）

1.创建/编辑初始化参数文件（initsales.ora）

spfile建不了，建pfile，找个现成的初始化参数文件改一改

cp initdb12c.ora initsales.ora

vim initsales.ora

所有db12c替换为sales

:1,$ s/db12c/sales/g

2.创建相应的目录结构

3.启动sales例程

echo $ORACLE\_SID

export ORACLE\_SID=sales

sqlplus / as sysdba

startup nomount

4.执行创建数据库的语句

创建数据文件和日志文件，控制文件会自动生成

system sysaux undotbs1 这三个表空间一定要在建库的时候建

create database sales

datafile '/u01/app/oracle/oradata/sales/system01.dbf' size 400m

sysaux datafile '/u01/app/oracle/oradata/sales/sysaux01.dbf' size 400m

undo tablespace undotbs1 datafile '/u01/app/oracle/oradata/sales/undotbs01.dbf' size 50m

默认的临时表空间也可以建一下：

default temporary tablespace temp tempfile '/u01/app/oracle/oradata/sales/temp01.dbf' size 20m

logfile

group 1 ('/u01/app/oracle/oradata/sales/redo01.log') size 10m,

group 2 ('/u01/app/oracle/oradata/sales/redo02.log') size 10m,

group 3 ('/u01/app/oracle/oradata/sales/redo03.log') size 10m;

5.创建数据字典视图

$ORACLE\_HOME rdbms admin catalog.sql

6.创建口令验证文件

$ORACLE\_HOME dbs orapwdb12c

orapwd file=orapwsales password=admin1#3

7.创建oracle内部包

脚本：$ORACLE\_HOME rdbms admin catproc

8.创建spfile

create spfile from pfile;

9.创建scott方案

脚本：share/scott.sql

10.加载产品用户概要文件

conn system/manager

11.配置监听器（服务器端）和服务名（客户端）

$ORACLE\_HOME network admin listener.ora

export DISPLAY=192.168.11.1:0.0图形界面

netmgr

监听程序：地址不变 ，数据库服务：添加数据库 sales SID=sales 保存

cat listener.ora 监听器多了sales

服务名中加sales 192.168.11.200 服务名：sales

12.配置EM Export

http://192.168.11.200:7531/em

* 控制文件

1.了解控制文件

2.查看控制文件的名称和位置

show parameter control\_files

select name from v$controlfile;

3.查看控制文件内容

select dbid,name from v$database;

oradebug tracefile\_name

4.控制文件的多路复用

5.创建控制文件

* 管理日志文件

1.了解日志文件

用来记录数据的变化，当系统或介质出问题提供恢复机制。

例程恢复，介质恢复

日志文件以组的方式组成，oracle中至少2组，每组中至少一个日志文件。

组内中若有多个成员则为镜像关系，避免损坏，组件循环方式使用。

一个文件写满了，进行日志切换（LGWR），当前文件序列号写到控制文件中，且设置检查点。

alter system switch logfile;发生日志切换时，会设置检查点

多选：日志切换时以下进程会工作LGWR，CKPT，DBWR，ARC

2.查询日志文件组信息（v$log）

select GROUP#,MEMBERS,SEQUENCE#,status from v$log;

出了问题知道哪个日志文件来恢复

select GROUP#,MEMBERS,SEQUENCE#,status,FIRST\_CHANGE#,NEXT\_CHANGE# from v$log;

3.查询日志文件成员信息（v$logfile）

desc v$logfile;

select GROUP#,MEMBER from v$logfile;

状态：active 恢复的时候还需要用到该日志文件

自动切换：写满

手动切换：alter system checkpoint;

4.添加日志文件组

应用场景：在生产系统中，出现日志文件等待事件，表示日志组不够了（大量事务来不及归档）。

修改数据库

alter database add logfile group 4 ('/u01/app/oracle/oradata/sales/redo04a.log') size 10m;添加新组，还在未使用状态

5.添加日志文件成员

alter database add logfile member '/u01/app/oracle/oradata/sales/redo04b.log' to group 4;镜像关系 不用写大小

6.删除日志文件组

组状态：current,active 均为使用状态，不能删。

数据库中只有两个组，不能删。

alter database drop logfile group 1; 使用组 不能删

alter system switch logfile; 切换， active不能删

alter system checkpoint; 可删

剩下的组内文件也要删 !rm /u01/app/oracle/oradata/sales/redo01.log

逻辑上删除了，物理上没删完，使用OMF改进

7.OMF（oracle manage file）

1）数据文件OMF：

不用omf：

select name from v$tablespace;

create tablespace tbs1 datafile '/u01/app/oracle/oradata/sales/tbs1.dbf' size 10m;

数据库创建文件时指定目的地：alter system set db\_create\_file\_dest='/u01/omf';

创建表空间create tablespace tbs1;

删除 drop tablespace tbs1;

2）日志文件OMF

日志文件会自动OMF：

alter database add logfile;在recovery\_area自动创

自动创/删组和组内文件。

OMF的组被删 对应文件也会删。

更改默认创建文件地址：

show parameter db\_create有几个属性写参数就创建几个文件

alter system set db\_create\_online\_log\_dest\_1='/u01/omf';

8.删除日志文件成员

组内只有一个成员，不能删。

current不能删，active可删。

9.清除日志文件组

clear：相当于删旧的加新的

10.清除日志文件成员

alter database clear logfile

12.设置数据库日志模式

select log\_mode from v$database;查看日志模式

改成归档模式 alter database archivelog 只能在mount阶段做，不能open做

archive log list查看模式

alter database open;回到open

13.查询归档日志信息（v$archived\_log）

select name from v$archived\_log;空的，还未归档

14.设置归档目的地并手工归档

否则放到闪回区中。

自动归档需要等写满。

show parameter log\_archive 日志归档目的地由参数控制

alter system set log\_archive\_dest\_1=''不能直接写/u01/arch/1 会报错 要写本地还是远程

alter system set log\_archive\_dest\_1='location=/u01/arch/1';本地

alter system set log\_archive\_dest\_2='location=/u01/arch/2';

archive log list查下一个需要归档的序号

alter system archive log current;产生两个一样的文件

select name from v$archived\_log;查看归档日志

show parameter log\_archive查看归档日志文件名格式log\_archive\_format：%t\_%s\_%r.dbf

修改格式：alter system set log\_archive\_format='%t\_%s\_%r.arc' scope=spfile; 重启生效

15.日志文件丢失的处理方法

1）非当前的丢失

select GROUP#,MEMBERS,SEQUENCE#,status,FIRST\_CHANGE#,NEXT\_CHANGE# from v$log;

select group#,member from v$logfile;

用操作系统删!rm /u01/app/oracle/oradata/sales/redo02.log

alter database clear logfile group 2;

clear就可以了

2）当前的丢失

clear不行

recover databse until cancel;

alter database open resetlogs;日志全部重建

archive log list 编号全部重新开始

* 管理表空间和数据文件

一个数据库有多个表空间，表空间不可以跨越数据库。

表空间由数据文件构成，数据文件不可以跨越表空间

段：存储结构

簇表：经常在一起使用的小表

嵌套表：表中还有表

区：空间分配单位

块：最小的读写单位

一个oracle块由多个OS块构成

1.小文件表空间和大文件表空间

自动创建为小文件表空间，一个表空间可以同时创多个数据文件

create tablespace users datafile '/u01/app/oracle/oradata/sales/users01.dbf' size 20m;

大文件表空间只有一个数据文件，无限大

2.表空间类型

1）常规表空间

2）撤销表空间(UNDO)

放回滚数据

create undo tablespace undotbs2 datafile'/u01/app/oracle/oradata/sales/undotbs2.dbf' size 20m;

alter system set undo\_tablespace=undotbs2;修改当前撤销表空间

3）临时表空间(TEMP)

排序

create temporary tablespace tempdef tempfile '/u01/app/oracle/oradata/sales/tempdef.dbf' size 20m;

alter database default temporary tablespace TEMPDEF;设置默认临时表空间

临时表空间组：

多用户访问一个表空间：争用

创建临时表空间组避免以上问题

select \* from dba\_tablespace\_groups;

组不需要创建，只需要添加即可

alter tablespace TEMP tablespace group group1;

create temporary tablespace temp1 tempfile '/u01/app/oracle/oradata/sales/temp1.dbf' size 10m tablespace group group1;

alter tablespace temp1 tablespace group group2;更改表空间

drop tablespace temp1;删临时表空间，临时表空间组内没有成员，表空间组会自动删除

alter tablespace temp tablespace group '';移除临时表空间

alter database default temporary tablespace group1;修改临时表空间组为默认临时表空间

3.查看表空间类型

desc dba\_tablespaces

select tablespace\_name,contents from dba\_tablespaces;

show parameter undo\_tablespace 查看当前撤销表空间

4.区

区分为数据字典管理和本地管理

本地管理表空间，区（数据文件）的开头有位图表示区号，1表示已被占用，0表示未占用

select tablespace\_name,extent\_management from dba\_tablespaces;

本地管理表空间优点：减少对数据字典的征用，没有回滚产生。

数据字典管理的更新属于事务，需要回滚。

碎片：数据不连续存放，文件大小不一样容易产生碎片。

create tablespace userdata datafile '/u01/app/oracle/oradata/sales/userdata01.dbf' size 10M extent management dictionary;

5.表空间状态

select tablespace\_name,status from dba\_tablespaces;

1）联机可读写

2）只读：（只改数据字典，不改物理）

create table scott.demo1(id int) tablespace tbs1;

insert into scott.demo1 values(100);

commit;

alter system checkpoint;

alter tablespace tbs1 read only;改为只读

create table scott.demo2(id int) tablespace tbs1;不能放

show parameter defer

alter system set deferred\_segment\_createion=true;

create table scott.demo2(id int) tablespace tbs1;能放了

insert into scott.demo1 values(200);不能插入

alter table scott.demo1 add name varchar(20);可加字段

drop table scott.demo1;可删

3）脱机

东西还在，不能访问

alter tablespace system offline;报错，系统表空间不能脱机

select tablespace\_name,contents,status from dba\_tablespaces;

sysaux可脱机。

sysaux:辅助系统表空间，增加了自动优化、管理的功能，需要系统数据辅助

10g之后增强了自动管理功能，system表空间负担过重，于是有关自优化的东西放到sysaux中。

可脱机：自优化功能没有了。

当前的撤销表空间不能脱机，非当前的可脱机。

临时表空间不能脱机。

出选择题：不能脱机的：system，当前的撤销表空间，临时表空间。

select tablespace\_name,file\_name from dba\_data\_files;

删表空间时，文件没有删，DDL数据定义语言，只改数据字典

删表空间时，表非空不能删。

drop tablespace tbs1 INCLUDING CONTENTS and datafiles;连内容及数据文件一块删

6.扩展表空间

1）修改原数据文件大小

select file\_name from dba\_data\_files where tablespace\_name='USERS';

①手工扩展

alter database datafile '...' resize 30m;

②自动扩展

alter database datafile '...' autoextend on next 5m maxsize 100m;每次用完+5M直到100M报错

alter database datafile '...' autoextend off;自动扩展关闭

2）添加新的数据文件

alter tablespace users add datafile '...' size 20m;

3）移动 重命名

alter database rename file '...' to '...'

7.表空间误删，没有做备份时恢复

用OS删掉,把误删的文件脱机 打开数据库先让人用，然后修复

alter database datafile 9 offline;

alter database open;

alter database create datafile 9;

recover datafile 9;

alter database datafile 9 online;

* Oracle安全

3A

1.验证

验证用户

1）sys用户

数据库不启动都能连

①操作系统验证

conn yyy/aaa as sysdba 随便写都能连，不安全

②口令文件验证

需要口令验证文件

cd $ORACLE\_HOME/dbs

orapwd file=orapwsales password=admin1#3 force=y

show parameter remote 检查初始化参数

remote\_login\_passwordfile

value:NONE,EXCLUSIVE,SHARED

NONE:禁用口令验证文件

EXCLUSIVE启用口令验证文件，单例程多用户

SHARED启用，多例程单用户

查哪些用户可连：v$pwfile\_users

select username,sysdba,sysoper,sysasm,sysbackup,sysdg,syskm from v$pwfile\_users;

sysdba:访问整个oracle服务器 包括instance database

sysoper：系统操作语言 管理instance

sysasm：可配置自动存储管理

sysbackup：只管用户的备份恢复操作

sysdg：dataguard，双机热备，数据库多份

syskm：数据库需要做加密时

修改sqlnet.ora：sysdba不写用户名、口令无法连

SQLNET.AUTHENTICATION\_SERVICES=(NONE)

口令文件是二进制文件 不能直接改：

grant sysoper to scott;

revoke sysoper from scott;

2）non-sys用户

①数据库验证

②操作系统验证

show parameter os\_authent\_prefix

数据库用户要对操作系统用户加前缀ops$

alter system set os\_authent\_prefix='' scope=spfile;

shutdown immediate

startup

③全局验证

基于LDAP

企业中有多个数据库，连到数据库需要使用其的用户。全局验证解决这个问题，映射，全局服务器。

2.授权

用户权限：系统权限，对象权限

1）系统权限

可让用户执行特定的操作

2）对象权限

可让用户操作或访问某个对象

选择题：选择系统权限/对象权限

3）权限传递原则

①对象权限是连带的，eg：

给权限grant：a->b->c

收回权限revoke：a->xb

Q：c？ A：c也没有权限了，a不能越级revoke

grant select ... on b with grant option;给与赋予对象权限的权力

②系统权限不连带

grant：a->b->c

revoke：a->xb

A：c还有系统权限，a可以越级revoke

grant create ... on b with grant system;给与赋予系统权限的权力

4）角色

定义：权限的集合，用一个角色组织多个权限。

简化权限管理：权限->角色->用户

动态权限管理：权限放在角色中，角色需要等待激活。

①自定义角色

create role r1;

create role r2 identified by r2;带口令的角色

grant select on scott.emp to r1,r2;

grant r1 to a;把r1角色指派给a用户（r1成为a的默认角色）

grant r2 to b;把r2角色指派给b用户（此时ab都还不能访问）

用户获得角色中的权限需要激活。

默认角色，登录时激活。非默认角色，需要显式激活。

alter user a default role none;a用户修改为无默认角色

set role r1;激活了 可访问了

带口令的角色登录时不能默认激活

set role r2 identified by r2;激活带口令的角色。

②预定义角色（系统预定义的）

role：connect，dba，resource，schedule\_admin，select\_catalog\_role（scott无法查询数据字典，赋予该权限才可以eg:desc dba\_tables）

③应用程序角色

通过应用程序激活，与上两种不同的是，不需要先将角色指派给用户，直接执行程序即可使用。

如何创建：

create role app\_r1 identified using scott.p1;

grant select on scott.emp to app\_r1;

create or replace procedure scott.p1 创建存储过程

authid current\_user as

begin

dbms\_session.set\_role('APP\_R1');

end;

grant execute on scott.p1 to a,b;使用户a,b可执行该过程

5）利用权限分析来实现最小权限原则

最小权限原则：需要用户做某件事时，只给能做这件事的最小权限。

如何确定最小权限：12c之前需要管理员判断，12c提供了包DBMS\_PROVILEGE\_CAPTURE。

开启捕获，用户操作，关闭捕获，生成报告，分析报告，得知用户使用了哪些权限，产生角色，将角色给用户。

EG：

create user priv\_test identified by test;

grant dba,resource to priv\_test;

创建捕获

begin

DBMS\_PRIVILEGE\_CAPTURE.create\_capture(...);

end;

启用捕获

begin

DBMS\_PRIVILEGE\_CAPTURE.enable\_capture(...);

end;

用户执行操作。。。

禁用捕获并产生分析结果

begin

DBMS\_PRIVILEGE\_CAPTURE.disable\_capture(...);

end;

begin

DBMS\_PRIVILEGE\_CAPTURE.generate\_result(...);

end;

查权限

select username,sys\_priv from dba\_used\_sysprivs where capture='dva\_res\_user\_pol' order by username,sys\_priv;

。。。看录屏吧，0331 145031-419

创建角色，将查到的权限指派给角色，角色指派给用户，之前指派的角色删掉。

6）概要文件（录屏0331 152012-795）

某用户登录后系统特别卡：该用户做了很多消耗资源的操作，做一个概要文件（口令，资源）限制他。

有效期42天：如果口令满足复杂度要求，暴力破解口令需要40天左右。

保留口令数5：五次修改口令后才能重复以前的口令。

复杂性函数：口令要求。函数放在sys下

失败登陆次数：超过次数会被锁住。

连接时间120:2h之后数据库自动断开。

组合限制：cpu 连接时间 内存 io

3.审核

监视数据库中可疑的活动。

1）强制审核（默认审核）

系统设置的审核，信息记录在警告日志文件中。

2）标准数据库审核

用户设置的。

①启用审核（初始化参数）

show parameter audit\_trail （参数值：extended xml none os db）

none：禁用审核，其他都是启用。db：审核记录保存在数据库的表（aud$）中。os：审核记录放在操作系统文件（查参数audit\_file\_dest）中。

xml：通过程序访问审核记录。extended：不可单用，要与其他参数值一起用，表示审核信息中包含其他详细信息，开销大。

alter system set audit\_trail='...' scope=spfile

②指定审核选项（noaudit ....取消审核）

a)用户审核（权限审核）：某用户行使某种权限时被记录下来。

audit select any table by scott; scott用户行使select权限会被记录下来

b)对象审核

audit delete on scott.emp whenever not successful; 只记录删失败 只要删该表中数据都会记录下来，不管是否删成功

audit delete on scott.emp whenever successful; 只记录删成功

audit delete on scott.emp whenever not successful; 只记录删失败

audit session whenever not successful; 只记录用户登录失败

audit update on scott.emp by session; 做同样的操作只记一次

audit update on scott.emp by access; 做同样的操作每次都记

c)语句审核：只审核一种行为

audit create trigger; 只要使用触发器都会记录

此时开启了后台进程的监视。

③查找审核信息

desc aud$ （不查表 查视图↓）

desc dba\_audit\_trail

TIMESTAMP USERNAME ACTION\_NAME

④维护审核记录

3）基于值的审核

上一点只记录了动作，没记录修改值。

利用触发器实现。一般不放在sys下。

与存储过程区别：存储过程可手工调用，触发器只能自动调用，满足某条件自动执行代码。

①DML触发器（DML语句）

create table scott.tr1\_test(a varchar(30));

create or replace trigger scott.tr1

after update on scott.emp

begin

insert into scott.tr1\_test values('Be Changed!');

end;

/

update emp set sal=2800 where empno=7369;

select \* from scott.tr1\_test;

rollback;

create or replace trigger tr1

after update on emp referencing old as o new as n for each row逐行触发，旧值用o调用，新值用n

begin

insert into sal\_change values(:o.ENAME,:o.sal,:n.sal);

end;

②系统触发器：触发动作为某一系统事件

除create trigger权限，还需要 grant administer database trigger to scott;

EG1：登录数据库

create or replace trigger tr1

after logon on database

begin

if user in('SYS','SYSTEM') then null;

else

insert into scott.logon\_rec values(user,sysdate);

end if;

end;

/

EG2：开启数据库:

create or replace trigger tr1

after startup on database

begin

sys.dbms\_shared\_pool.keep('SCOTT.P1');

end;

/

grant execute on dbms\_shared\_pool to scott;

重启

* 数据移动 录屏：200403

sqlldr：

1.数据源（数据文件）

2.坏文件：保存坏数据的文件（坏数据：格式不对。eg字段设置为int 但数据不为int）

3.丢弃文件：可选的，不指定不产生。放数据格式没问题，但不符合条件的（eg不符合设置的范围）。

4.日志文件：出问题查日志文件。

5.控制文件：加载sqlldr。

待指定：

1）load

2）数据源（infile）

如果没有专门的数据文件，数据直接写在控制文件内。

3）目标表（into table）

insert into：插入空表

append into：追加记录

relace into：表先清空（delete）再加

truncate into：表先清空（truncate）再加

4）数据格式

①记录结束：

默认为一行一条记录。

如果为2行一条：concatenate 2

如果记录不为固定行数（有2有3）：continueif this （1:1） = ‘-’//出现-表示新纪录开始

②字段结束：

terminated by “，”//遇到，表示字段结束

字段类型：

字符型只有char，日期类型date[“format”] eg date “dd-mon-yyyy”，integer external整数

decimal external 小数，zoned（a，b）a为总长度、b为小数点后多少位。

③缺失数据：

trailing nullcols：允许尾部为空记录。

optionlly enclosed by ‘ ”’：可选定界符。

* 导入导出工具exp/imp

导出：emp：exp scott/tiger file=/u01/exp/emp2.dmp tables=emp,dept

导出整个数据库 emp username/password file=xxx tables=xxx full=y

导出某用户下所有表 emp username/password file=xxx tables=xxx owner=username1

导出表的子集 emp username/password file=xxx tables=xxx query='where ...'

用参数文件导 exp parfile=p1.par

导入：imp：imp username/password file=xxx tables=xxx

查用法 exp help=y，imp help=y

* 导入导出工具expdp/impdp

采用数据泵：并行机制提高速度（大任务，足够多的空闲资源）

先在数据库下创建逻辑目录 create directory expdp1 as '/u01/expdp1'

grant read,write on directory expdp1 to scott

导出：expdp username/password directory=expdp1 file=xxx tables=xxx

查用法 expdp/impdp help=y

1）外部数据文件——>数据库

2）数据库——>数据库

3）数据库——>数据文件

导入其他用户的表可能权限不够：使用角色EXP\_FULL\_DATABASE

放在sys下的表被认为为系统数据，导不出来。

* 外部表

不在数据库中，在操作系统文件。读写操作系统文件，不增加数据库负担。

文本文件oracleldr

二进制文件 pump

1.产生外部表定义

2.指定驱动类型和默认目录

3.知道访问参数

1）产生记录 records delimited by

2）产生字段

missing fiedl values are null 没写就为空

4.指定字段顺序

5.指定数据源

* 备份恢复（录屏200407）

最小化数据丢失。

1.错误种类

语句错误，用户进程错误（有进程监视器进程去管理，dba不用管），网络故障（冗余机制：配多个网卡/备用监听），

用户错误（用户误操作导致数据丢失，通过闪回修复，but闪回到什么点？，使用用户挖掘器Oracle LogMiner查日志文件）：

2.使用LogMiner

1）启用日志补充文件===>日志本来记得没有那么详细

select SUPPLEMENTAL\_LOG\_DATA\_MIN from v$database;

alter database add SUPPLEMENTAL LOG DATA;

2）产生数据字典文件dbms\_logmnr\_d

EXECUTE dbms\_logmnr\_d.build('v816dict.ora','DICT');

3）开始一个事务

4）添加需要分析的日志文件

select group#,status from v$log;

select member from v$logfile where group#=4；(选active)

EXECUTE dbms\_logmnr.add\_logfile(LogFileName=>'/u01/app/oracle/oradata/sales/redo03.log',Options=>dbms\_logmnr.new) （第一个.new 以后.addfile）

5）启动分析

EXECUTE dbms\_logmnr.start\_logmnr(DictFileName=>'/u01/dict/v816dict.ora');

6）查询v$logmnr\_contents

select TIMESTAMP,USERNAME,SQL\_REDO from v$logmnr\_contents where SEG\_NAME='EMP';（注意写在引号中需要大小写）

例程崩溃（系统管理器进程），

3.备份

介质恢复（Media Recovery）

分类：

冷备份：关闭数据库的备份（一致性备份/脱机备份）

热备份：打开数据库的备份（联机备份）

用户管理的备份，服务器管理的备份（RMAN：Recover Manage）

全库备份（完整的数据库备份），部分数据库备份

全备份（表空间对应的数据文件全部备份），增量备份（变化的部分备份）

4.用户管理的备份和恢复

1）冷备份

a.处于shutdown状态

b.做文件列表，保证文件不丢失

数据库启动过程：初始化参数文件，控制文件，数据文件，日志文件，（口令文件可选）。

spool /u01/salesbak/filelist.txt

show parameter spfile

select name from v$controlfile;

select member from v$logfile;

select name from v$datafile;

select name from v$tempfile;

c.创建测试表

create table scott.cold(a varchar(30)) tablespace users;

insert into scott.cold values('Before Backup');commit;

alter system checkpoint;

d.插入新记录

insert into scott.cold values('After Backup');

e.破坏数据库

f.恢复数据库

1）还原初始化参数文件

2）创建相应的目录结构

3）启动例程

4）还原控制文件

5）加载数据库

6）还原数据文件和日志文件

7）恢复数据文件 recover database;

8）打开数据库

9）检查数据

5.热备份

1）归档模式

archive log list

2）备份模式

用户可能一直在修改数据，插一个检查点，备份那时的数据。

select \* from v$backup;

备份user表空间下的表

select tablespace\_name from dba\_tables where table\_name='COLD';

select file\_name from dba\_data\_files where tablespace\_name='USERS';

select file#,name from v$datafile; 5 6号

alter tablespace users begin backup;

3）备份数据文件

4）插入新记录

5）结束备份模式

不然不能脱机

6）破坏数据文件

7）还原数据文件

8）恢复数据文件

recover tablespace users;

9）检查数据

整库备份：alter database begin backup;

做完alter database end backup;

6.服务器管理的备份（RMAN）

1）认识RMAN

mount阶段

rman连接时需要指定数据库类型（目标数据库，目录数据库，辅助数据库）

目标：需要被备份的数据库

目录：各个目标数据库的控制文件中的相关文件

辅助：eg：用户做了误操作（delete），数据库还原完后recover，这个delete还会再做一遍，克隆一个辅助库，在该库将数据恢复，

然后再将辅助数据库内的数据导入。

connect target /

backup tablespace users;

2）rman常用命令：200410

report，list，delete，backup，restore，recover，copy，show，configure，crosscheck，catalog

backup：备份初始化参数文件：backup spfile; 备份数据库：backup database;

rman中可直接执行sql语句

list backup; 查看之前做的备份

list copy of database;查看之前拷贝的

拷贝copy datafile 4 to '....dbf';

delete noprompt backup;删除备份

delete noprompt copy of database;删除拷贝

crosscheck copy 交叉检查

备份默认放在omf 闪回区中

rman的配置命令：show all;

report obsolete;查看有无过期备份

report need backup;查看需要做的备份

配置命令configure

CONFIGURE RETENTION POLICY clear;修改回默认值

CONFIGURE RETENTION POLICY to recovery window of 7 days;备份保留七天

CONFIGURE BACKUP OPTIMIZATION 备份前比较一下有没有改变，如果此开关开了，表示比较一下，变化了再备份，不变化不备份。

如果数据库中存在多个只读表空间，则可打开。

CONFIGURE CONTROLFILE AUTOBACKUP 打开的话 不管做什么备份操作都会备份一遍控制文件。

概念：channel通道，配两头，要备份什么，及要备份去哪。

configure channel device type disk format '/u01/salesbak/%d\_%u\_%T';修改备份文件存储地址,注意不要写死,不然不能重复备份

CONFIGURE DEVICE TYPE DISK PARALLELISM 3开三个通道备份

全备份：spfile controlfile logfile(联机 归档) datafile

alter system archive log current;

以上为**完全恢复**

6.**不完全恢复**

完全恢复：日志中做的完全恢复出来。

做不完全恢复的情况：

1）被动：日志丢掉了

2）主动：之前做了错误操作被记入日志。

用备份的控制文件恢复：drop删掉了表空间，当前控制文件中就没有这个表空间了。

过程：shutdown，restore，mount，recover，open（resetlogs）

7.做辅助数据库恢复

操作有为对、错、对，则不能推到某个点，要做辅助数据库。

select member from v$logfile;

select name from v$datafile;

* 表空间基于时间点的恢复200414

create tablespace tbs1 datafile '/u01/app/oracle/oradata/sales/tbs101.dbf' size 10m;

create table scott.t1 tablespace tbs1 as select \* from scott.emp;

create index scott.idx

select systimestamp from dual;

建表1，表2索引表1，恢复表1不能恢复表2，要同时恢复。

* 目录数据库（又称恢复目录数据库）

可选的存储，包含备份集和备份片。

list backup：BS（backup set 备份集），BP（backup piece备份片），存放有关备份的东西。

1.使用恢复目录的情况

1）集中存储多个目标数据库中关于备份和恢复的系统数据

2）存放一个目标数据库的多个版本的原数据：incarnation（化身：存一个数据库的多个版本）

list incarnation;

reset database to incarnation 2;要在mount状态，该操作可以将数据库恢复到resetlogs之前

3）存储RMAN的脚本

2.创建恢复目录

db12c：目录数据库

sales，oradup：目标数据库

1）创建用来保存备份和恢复系统数据的表空间

2）创建目录所有者（用户）

create user catowner identified by admin;

3）修改目录所有者的默认表空间

alter user catowner default tablespace cat;

4）赋予相应的权限

grant connect,recovery\_catalog\_owner to catowner;

alter user catowner quota unlimited on cat;使用权无限

5）rman连接到目录数据库

rman catalog catowner/admin

6）创建恢复目录

create catalog tablespace 'CAT';

7）连接到目标数据库

connect target sys/admin1#3@sales

8）注册目标数据库

9）手工同步恢复目录

3.RMAN脚本

* 闪回技术

传统的恢复比较慢，闪回快而简单，可以闪回逻辑错误，不能闪回物理错误。

1.闪回数据库

1）前提：

数据库为归档模式，启用闪回日志。

2）和闪回数据库相关的初始化参数：

show parameter recover;

db\_recovery\_file\_dest闪回日志保存地点（闪回区）

db\_recovery\_file\_dest\_size闪回区大小

show parameter retention

db\_flashback\_retention\_target闪回日志保留时间，值的单位为分钟

动态性能视图：v$flashback\_database\_log

select current\_scn from v$database;

做破坏：drop user scott cascade;

flashback database to scn nnnn;在mount阶段

alter database open resetlogs;

2.闪回表(dml)

delete

1）前提

激活表的行移动特性

select table\_name,ROW\_MOVEMENT from user\_tables;

alter table emp enable row movement;

set time on

flashback table emp to timestamp to\_timestamp('2020-04014 16:44:58','yyyy-mm-dd hh24:mi:ss');

3.闪回删除ddl

drop操作

回收站

4.闪回查询

闪回查询表 闪回版本查询 闪回事务查询

* CDB容器数据库200417考：非CDB与CDB区别

原来所有系统数据（包括oracle 用户）都放在system表空间里，每个数据库都一个instance。

目的：用户只管数据，oracle管系统，在cdb中：只有一个instance，可以有多个数据库，即只有一个初始化参数文件。

oracle系统数据，放在cdb下用户系统表空间。

用户系统数据，放在pdb下。即各自数据库中

只有一个instance，即只有一个初始化参数文件。即控制文件只放在根下，pdb中没有。

种子数据库总以只读方式打开，是一个容器，不可以被删掉。

PDB可插拔数据库

* 用户

公共用户：整个cdb范围内可见

本地用户：仅在pdb可见

公共权限可给公共用户，不能给本地用户。

本地权限都可给。

本地角色可给本地用户，公共角色可给本地、公共用户。

本地角色可给本地角色，公共角色可给本地、公共角色。

考试：公共、本地间 用户 权限 角色的指派。