

**期末项目设计报告**

|  |  |  |  |
| --- | --- | --- | --- |
| 题 目 | 基于Oracle的学生成绩管理的数据库设计 | | |
| 课程 | Oracle数据库应用 | | |
| 学 院 | 信息科学与工程学院 | | |
| 专 业 | 软件工程 | 年级 | 2018级 |
| 学生姓名 | 马勇 | 学号 | 201811606228 |
| 指导教师 | 赵卫东 | 职称 | 副教授 |

|  |  |  |  |
| --- | --- | --- | --- |
| **评分项** | **评分标准** | **满分** | **得分** |
| 文档整体 | 文档内容详实、规范，美观大方 | 10 |  |
| 表设计 | 表，表空间设计合理，数据合理 | 20 |  |
| 用户管理 | 权限及用户分配方案设计正确 | 20 |  |
| PL/SQL设计 | 存储过程和函数设计正确 | 30 |  |
| 备份方案 | 备份方案设计正确 | 20 |  |
| **得分合计** | | |  |

2021 年 6 月 1 日

目录

[一、前言 3](#_Toc74671733)

[二. 需求分析 3](#_Toc74671734)

[三、数据库设计 4](#_Toc74671735)

[1.数据库结构设计 4](#_Toc74671736)

[2.实体联系图 4](#_Toc74671737)

[3.数据库源代码 5](#_Toc74671738)

[（1）建表 5](#_Toc74671739)

[（2）存储过程 6](#_Toc74671740)

[四．操作步骤 16](#_Toc74671741)

[1、创建角色role\_1和用户user\_1，并授权和分配空间 16](#_Toc74671742)

[2、创建角色role\_2和用户user\_2，并授权和分配空间 16](#_Toc74671743)

[3、创建表 17](#_Toc74671744)

[4、创建存储过程 19](#_Toc74671745)

[5、插入数据 19](#_Toc74671746)

[6、全数据库备份 21](#_Toc74671747)

[（1）开始备份 21](#_Toc74671748)

[（2）查看备份文件内容 22](#_Toc74671749)

[（3）备份后修改数据 23](#_Toc74671750)

[（4）删除数据库文件，模拟数据库文件损坏 23](#_Toc74671751)

[（5）删除数据库文件后修改数据 24](#_Toc74671752)

[（6）数据库完全恢复 24](#_Toc74671753)

[（7）设计容灾方法，使用两台主机，通过DataGuard实现数据库整体的异地备份 25](#_Toc74671754)

# 一、前言

学生成绩管理是一个学校必不可少的部分，随着计算机和计算机知识的普及，学生成绩管理系统得到了更大的发展空间，通过对学生成绩管理系统的开发，可以提高校务人员的工作效率。

学生成绩管理是一个学校不可缺少的部分，一个良好的学生成绩管理系统应该能够为用户提供充足的信息和快捷的查询手段。学生成绩管理系统对学校加强学生成绩管理有着极其重要的作用.由于各个大学都在持续扩招，学生的数量日益庞大，对于如何管理如此庞大的学生成绩数据显得更为复杂，传统的手工成绩管理不仅工作量大，而且容易出现问题，如：效率低、保密性差，另外时间一长，将产生大量的文件和数据，这对于查找、更新和维护都带来了不少的困难。已不能适应时代的发展。

随着科学技术的不断提高，计算机科学日渐成熟，其强大的功能已为人们深刻认识，它已进入人类社会的各个领域并发挥着越来越重要的作用。

作为计算机应用的一部分，使用计算机对学生成绩信息进行管理，具有手工管理所无法比拟的优点。例：检索迅速、查找方便、可靠性高、存储量大、保密性好、寿命长、成本低等。这些优点能够极大地提高管理者管理的效率,也是学校走向科学化、正规化管理,与世界接轨的重要条件。

# 二. 需求分析

1. 学生基本信息及所选科目成绩的录入或导入。
2. 基本信息的查询（分系、班级；分科目）与修改。
3. 对每系或每班各科成绩进行分析（即求单科平均成绩、及格率和优秀率）；
4. 对所开课程的成绩分析（求其平均成绩，最高分和最低分）；
5. 对学生考试成绩进行排名（单科按系别或班级进行排名，对每一个班级，同一学期学生总体成绩进行排名，并显示各科成绩信息）
6. 不及格情况的统计、导出、打印

# 三、数据库设计

1.数据库结构设计

院系表（cs）

|  |  |  |  |
| --- | --- | --- | --- |
| 字段名 | 数据类型 | 是否为空 | 说明 |
| Cs\_class | Varchar2(50) | Not null | 班级名（主键） |
| Cs\_dept | Varchar2(50) | Not null | 系别 |
| Cs\_grade | Varchar2(50) | Not null | 年级 |

成绩表（gd）

|  |  |  |  |
| --- | --- | --- | --- |
| 字段名 | 数据类型 | 是否为空 | 说明 |
| Gd\_stu\_no | Varchar2(15) | Not null | 学号（联合主键、外键） |
| Gd\_sub\_subject | Varchar2(50) | Not null | 科目（联合主键、外键） |
| Gd\_grade | Number(3) | Not null | 成绩 |

科目表（sub）

|  |  |  |  |
| --- | --- | --- | --- |
| 字段名 | 数据类型 | 是否为空 | 说明 |
| Sub\_subject | Varchar2(50) | Not null | 科目名（主键） |
| Sub\_teacher\_name | Varchar2(10) |  | 任课老师名 |

学生表（student）

|  |  |  |  |
| --- | --- | --- | --- |
| 字段名 | 数据类型 | 是否为空 | 说明 |
| Stu\_no | Varchar2(15) | Not null | 学号(主键) |
| Stu\_name | Varchar2(10) | Not null | 学生名 |
| Stu\_cs\_class | Varchar2(50) |  | 班级名（外键） |
| Stu\_sex | Varchar(4) | Not null | 性别(约束) |
| Stu\_birthday | Varchar2(50) |  | 出生日期 |
| Stu\_phone | Varchar2（200） |  | 联系电话 |

用户表（user\_）

|  |  |  |  |
| --- | --- | --- | --- |
| 字段名 | 数据类型 | 是否为空 | 说明 |
| name | Varchar2(50) | Not null | 用户名(主键) |
| Passwd | Varchar2(50) | notnull | 密码 |

2.实体联系图

学生信息

成绩信息

教师用户

管理

班级信息

科目信息

管理

管理

管理

3.数据库源代码

（1）建表

--创建用户表

create table user\_(name varchar(50) primary key,passwd varchar(50) not null);

--创建院系表

create table cs(

cs\_class varchar(50) primary key,

cs\_dept varchar(50) not null,

cs\_grade varchar(50) not null

);

--创建学生表

create table student(

stu\_no varchar(15) primary key,

stu\_name varchar(10) not null,

stu\_sex varchar(4) not null check (stu\_sex='男'or stu\_sex='女'),

stu\_birthday varchar(50) ,

stu\_phone varchar(50),

stu\_cs\_class varchar(50),

constraint class\_fk foreign key(stu\_cs\_class) references cs(cs\_class)

on delete cascade

);

--创建科目表

create table sub(

sub\_subject varchar(50) primary key,

sub\_teacher\_name varchar(10)

);

--创建成绩表

create table gd(

gd\_stu\_no varchar(15),

gd\_sub\_subject varchar(50),

gd\_grade number(3) not null,

primary key(gd\_stu\_no,gd\_sub\_subject),

constraint fk\_gd\_stu\_no foreign key(gd\_stu\_no) references student(stu\_no),

constraint fk\_gd\_sub\_subject foreign key(gd\_sub\_subject) references sub(sub\_subject)

);

（2）存储过程

create or replace procedure p\_insert\_stu(

p\_stu\_no in student.stu\_no%type,

p\_stu\_name in student.stu\_name%type,

p\_stu\_sex in student.stu\_sex%type,

p\_stu\_birthday in student.stu\_birthday%type,

p\_stu\_phone in student.stu\_phone%type,

p\_stu\_cs\_class in student.stu\_cs\_class%type,

s\_state\_values out number

) as

begin

insert into student (stu\_no,stu\_name,stu\_sex ,stu\_birthday,stu\_phone,stu\_cs\_class)

values (p\_stu\_no,p\_stu\_name,p\_stu\_sex,p\_stu\_birthday,p\_stu\_phone,p\_stu\_cs\_class);

if sql%rowcount=1

then s\_state\_values:=0;

end if;

exception

when others then s\_state\_values:=1;

commit;

end ;

/

create or replace procedure c\_insert\_stu(

c\_stu\_no in student.stu\_no%type,

c\_state\_values out number)

as

begin

select count(\*) into c\_state\_values from student

where stu\_no=c\_stu\_no;

end ;

/

create or replace procedure c\_insert\_stu\_fk(

c\_cs\_class in cs.cs\_class%type,

c\_state\_values\_fk out number)

as

begin

select count(\*) into c\_state\_values\_fk from cs

where cs\_class=c\_cs\_class;

end ;

/

create or replace procedure p\_insert\_cs(

p\_cs\_dept in cs.cs\_dept%type,

p\_cs\_grade in cs.cs\_grade%type,

p\_cs\_class in cs.cs\_class%type,

s\_state\_values out number)

as

begin

insert into cs(cs\_dept,cs\_grade,cs\_class)

values(p\_cs\_dept, p\_cs\_grade, p\_cs\_class);

if sql%rowcount=1

then s\_state\_values:=0;

end if;

exception

when others then s\_state\_values:=1;

commit;

end ;

/

create or replace procedure c\_insert\_cs(

c\_cs\_class in cs.cs\_class%type,

c\_state\_values out number)

as

begin

select count(\*) into c\_state\_values from cs

where cs\_class=c\_cs\_class;

end ;

/

create or replace procedure p\_insert\_sub(

p\_sub\_subject in sub.sub\_subject%type,

p\_sub\_teacher\_name in sub.sub\_teacher\_name%type,

s\_state\_values out number)

as

begin

insert into sub(sub\_subject,sub\_teacher\_name)

values(p\_sub\_subject, p\_sub\_teacher\_name);

if sql%rowcount=1

then s\_state\_values:=0;

end if;

exception

when others then s\_state\_values:=1;

commit;

end;

/

create or replace procedure c\_insert\_sub(

c\_sub\_subject in sub.sub\_subject%type,

c\_state\_values out number)

as

begin

select count(\*) into c\_state\_values from sub

where sub\_subject=c\_sub\_subject;

end ;

/

create or replace procedure p\_insert\_gd(

p\_gd\_stu\_no in gd.gd\_stu\_no%type,

p\_gd\_sub\_subject in gd.gd\_sub\_subject%type,

p\_gd\_grade in gd.gd\_grade%type,

s\_state\_values out number)

as

begin

insert into gd(gd\_stu\_no,gd\_sub\_subject,gd\_grade)

values(p\_gd\_stu\_no,p\_gd\_sub\_subject,p\_gd\_grade);

if sql%rowcount=1

then s\_state\_values:=0;

end if;

exception

when others then s\_state\_values:=1;

if p\_gd\_grade=null

then s\_state\_values:=2;

end if;

commit;

end ;

/

create or replace procedure c\_insert\_gd(

p\_gd\_stu\_no in gd.gd\_stu\_no%type,

p\_gd\_sub\_subject in gd.gd\_sub\_subject%type,

c\_state\_values out number)

as

begin

select count(\*) into c\_state\_values from (

select gd\_stu\_no from gd

where gd\_sub\_subject=p\_gd\_sub\_subject

)

where gd\_stu\_no=p\_gd\_stu\_no;

end ;

/

create or replace procedure c\_insert\_gd\_fk1(

c\_stu\_no in student.stu\_no%type,

c\_state\_values\_fk1 out number)

as

begin

select count(\*) into c\_state\_values\_fk1 from student

where stu\_no=c\_stu\_no;

end ;

/

create or replace procedure c\_insert\_gd\_fk2(

c\_sub\_subject in sub.sub\_subject%type,

c\_state\_values\_fk2 out number)

as

begin

select count(\*) into c\_state\_values\_fk2 from sub

where sub\_subject=c\_sub\_subject;

end ;

/

create or replace procedure p\_select\_stu\_no\_info(

p\_stu\_no in out student.stu\_no%type,

p\_stu\_name out student.stu\_name%type,

p\_stu\_sex out student.stu\_sex%type,

p\_stu\_birthday out student.stu\_birthday%type,

p\_stu\_phone out student.stu\_phone%type,

p\_stu\_cs\_class out student.stu\_cs\_class%type,

p\_cs\_dept out cs.cs\_dept%type,

p\_cs\_grade out cs.cs\_grade%type,

c\_state\_values out number)

as

begin

c\_state\_values:=0;

select stu\_no, stu\_name,stu\_sex ,stu\_birthday ,stu\_phone ,stu\_cs\_class ,cs\_dept ,cs\_grade

into p\_stu\_no, p\_stu\_name, p\_stu\_sex, p\_stu\_birthday, p\_stu\_phone, p\_stu\_cs\_class, p\_cs\_dept, p\_cs\_grade

from student, cs

where stu\_cs\_class=cs\_class and stu\_no=p\_stu\_no;

exception

when NO\_DATA\_FOUND then c\_state\_values:=1;

when others then c\_state\_values:=2;

end ;

/

create or replace procedure p\_select\_stu\_no\_gd(p\_stu\_no in student.stu\_no%type , cursor\_gd out sys\_refcursor)

as

begin

open cursor\_gd for

select gd\_sub\_subject,gd\_grade

from gd

where gd\_stu\_no=p\_stu\_no;

end;

/

create or replace procedure p\_select\_stu\_name(

p\_stu\_name in student.stu\_name%type ,

cursor\_stu out sys\_refcursor,

c\_state\_values out number)

as

begin

c\_state\_values:=0;

open cursor\_stu for

select stu\_no, stu\_name,stu\_sex ,stu\_birthday ,stu\_phone ,stu\_cs\_class ,cs\_dept ,cs\_grade

from student, cs

where stu\_name=p\_stu\_name and stu\_cs\_class=cs\_class;

exception

when others then c\_state\_values:=2;

end;

/

create or replace procedure p\_pm\_sum(

cursor\_gd\_sum out sys\_refcursor

)

as

begin

open cursor\_gd\_sum for

select distinct sno,stu\_name,sum

from student,(select gd\_stu\_no as sno,sum(gd\_grade) as sum

from gd

group by gd\_stu\_no

)

where stu\_no=sno

order by sum desc;

end;

/

create or replace procedure p\_pm\_sub(

p\_gd\_sub\_subject in gd.gd\_sub\_subject%type,

cursor\_gd\_sub out sys\_refcursor

)

as

begin

open cursor\_gd\_sub for

select gd\_stu\_no,stu\_name,gd\_grade

from gd,student

where gd\_sub\_subject=p\_gd\_sub\_subject and stu\_no=gd\_stu\_no

order by gd\_grade desc;

end;

/

create or replace procedure p\_pm\_class(

p\_stu\_cs\_class in student.stu\_cs\_class%type,

cursor\_gd\_class out sys\_refcursor

)

as

begin

open cursor\_gd\_class for

select distinct sno,stu\_name,sum

from student,(select gd\_stu\_no as sno,sum(gd\_grade) as sum

from gd

group by gd\_stu\_no

)

where stu\_no=sno and stu\_cs\_class=p\_stu\_cs\_class

order by sum desc;

end;

/

create or replace procedure p\_fx(

cursor\_fx out sys\_refcursor

)

as

begin

open cursor\_fx for

select gd\_sub\_subject,max(gd\_grade) as zuigao,min(gd\_grade) as zuidi,avg(gd\_grade) as pingjun,SUM(CASE WHEN gd\_grade>=60 THEN 1 ELSE 0 END)/COUNT(\*)\*100 AS jigelv,

SUM(CASE WHEN gd\_grade>=80 THEN 1 ELSE 0 END)/COUNT(\*)\*100 AS youxiulv

from gd

group by gd\_sub\_subject;

end;

/

create or replace procedure p\_bujige(

cursor\_bujige out sys\_refcursor

)

as

begin

open cursor\_bujige for

select gd\_stu\_no,gd\_sub\_subject,gd\_grade

from gd

where gd\_grade<60;

end;

/

create or replace procedure p\_update\_stu(

p\_stu\_no in student.stu\_no%type,

p\_stu\_name in student.stu\_name%type,

p\_stu\_sex in student.stu\_sex%type,

p\_stu\_birthday in student.stu\_birthday%type,

p\_stu\_phone in student.stu\_phone%type,

p\_stu\_cs\_class in student.stu\_cs\_class%type,

s\_state\_values out number

) as

begin

update student

set stu\_name=p\_stu\_name,stu\_sex=p\_stu\_sex,stu\_birthday=p\_stu\_birthday,stu\_phone=p\_stu\_phone,stu\_cs\_class=p\_stu\_cs\_class

where stu\_no=p\_stu\_no;

if sql%rowcount=1

then s\_state\_values:=0;

end if;

exception

when others then s\_state\_values:=1;

commit;

end ;

/

create or replace procedure p\_delete\_stu(

p\_stu\_no in student.stu\_no%type,

s\_state\_values out number)

as

begin

delete from gd where gd\_stu\_no=p\_stu\_no;

delete from student where stu\_no=p\_stu\_no;

s\_state\_values:=0;

exception

when others then s\_state\_values:=1;rollback;

commit;

end;

/

create or replace procedure p\_update\_gd(

p\_gd\_stu\_no in gd.gd\_stu\_no%type,

p\_gd\_sub\_subject in gd.gd\_sub\_subject%type,

p\_gd\_grade in gd.gd\_grade%type,

s\_state\_values out number)

as

begin

update gd

set gd\_stu\_no=p\_gd\_stu\_no,gd\_sub\_subject=p\_gd\_sub\_subject,gd\_grade=p\_gd\_grade

where gd\_stu\_no=p\_gd\_stu\_no and gd\_sub\_subject=p\_gd\_sub\_subject;

if sql%rowcount=1

then s\_state\_values:=0;

end if;

exception

when others then s\_state\_values:=1;rollback;

commit;

end ;

/

create or replace procedure p\_delete\_gd(

p\_gd\_stu\_no in gd.gd\_stu\_no%type,

p\_gd\_sub\_subject in gd.gd\_sub\_subject%type,

s\_state\_values out number)

as

begin

delete from gd

where gd\_stu\_no=p\_gd\_stu\_no and gd\_sub\_subject=p\_gd\_sub\_subject;

if sql%rowcount=1

then s\_state\_values:=0;

end if;

exception

when others then s\_state\_values:=1;rollback;

commit;

end ;

/

create or replace procedure p\_denglu(

p\_name in user\_.name%type,

p\_passwd in user\_.passwd%type,

s\_state\_values out number

)

as

begin

select count(\*) into s\_state\_values

from user\_

where name=p\_name and passwd=p\_passwd;

end;

/

create or replace procedure p\_update\_passwd(

p\_name in user\_.name%type,

p\_passwd in user\_.passwd%type,

s\_state\_values out number

)

as

begin

update user\_

set passwd= p\_passwd

where name=p\_name;

if sql%rowcount=1

then s\_state\_values:=0;

end if;

exception

when others then s\_state\_values:=1;rollback;

commit;

end;

/

create or replace procedure p\_select\_no(

cursor\_stu\_no out sys\_refcursor

)

as

begin

open cursor\_stu\_no for

select stu\_no

from student

order by stu\_no;

end;

/

create or replace procedure p\_select\_all(

cursor\_all out sys\_refcursor

)

as

begin

open cursor\_all for

select stu\_no, stu\_name,stu\_sex ,stu\_birthday ,stu\_phone ,stu\_cs\_class ,cs\_dept ,cs\_grade

from student, cs

where stu\_cs\_class=cs\_class

order by stu\_no;

end;

/

create or replace procedure p\_select\_count(

myrowcount out number

)

as

begin

select count(\*) into myrowcount

from student;

end;

/

# 四．操作步骤

1、创建角色role\_1和用户user\_1，并授权和分配空间

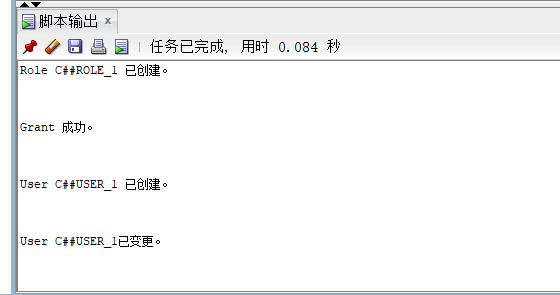
--创建角色role\_1和用户user\_1，并授权和分配空间

create role c##role\_1;

grant connect, resource, create view to c##role\_1;

create user c##user\_1 identified by 123 default tablespace users temporary tablespace temp;

alter user c##user\_1 quota 50m on users;



2、创建角色role\_2和用户user\_2，并授权和分配空间

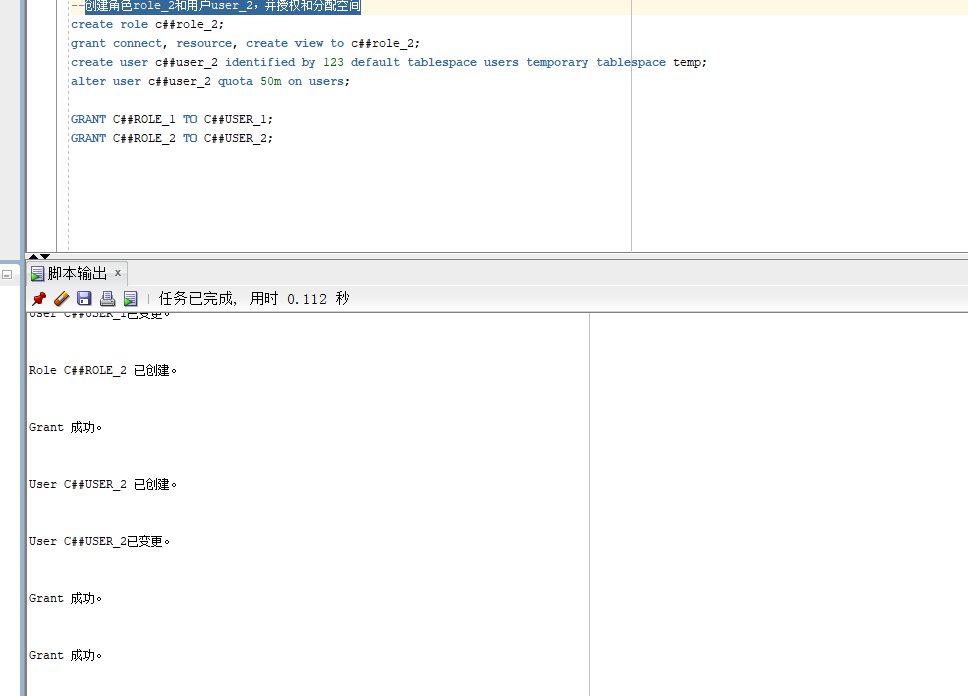
--创建角色role\_2和用户user\_2，并授权和分配空间

create role c##role\_2;

grant connect, resource, create view to c##role\_2;

create user c##user\_2 identified by 123 default tablespace users temporary tablespace temp;

alter user c##user\_2 quota 50m on users;



3、创建表

--创建用户表

create table user\_(name varchar(50) primary key,passwd varchar(50) not null);

--创建院系表

create table cs(

cs\_class varchar(50) primary key,

cs\_dept varchar(50) not null,

cs\_grade varchar(50) not null

);

--创建学生表

create table student(

stu\_no varchar(15) primary key,

stu\_name varchar(10) not null,

stu\_sex varchar(4) not null check (stu\_sex='男'or stu\_sex='女'),

stu\_birthday varchar(50) ,

stu\_phone varchar(50),

stu\_cs\_class varchar(50),

constraint class\_fk foreign key(stu\_cs\_class) references cs(cs\_class)

on delete cascade

);

--创建科目表

create table sub(

sub\_subject varchar(50) primary key,

sub\_teacher\_name varchar(10)

);

--创建成绩表

create table gd(

gd\_stu\_no varchar(15),

gd\_sub\_subject varchar(50),

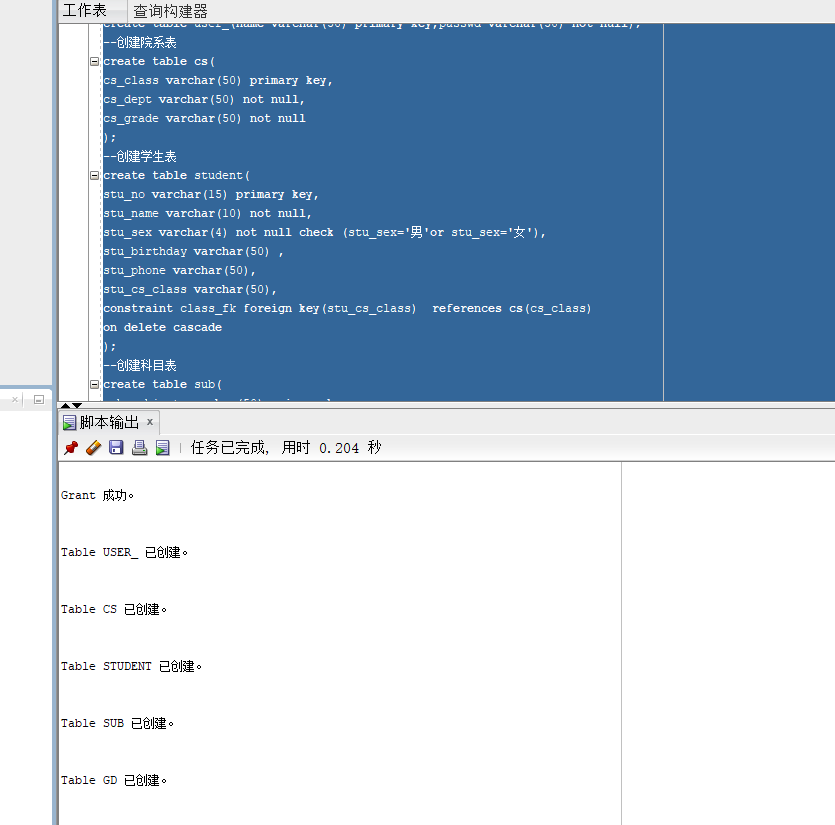
gd\_grade number(3) not null,

primary key(gd\_stu\_no,gd\_sub\_subject),

constraint fk\_gd\_stu\_no foreign key(gd\_stu\_no) references student(stu\_no),

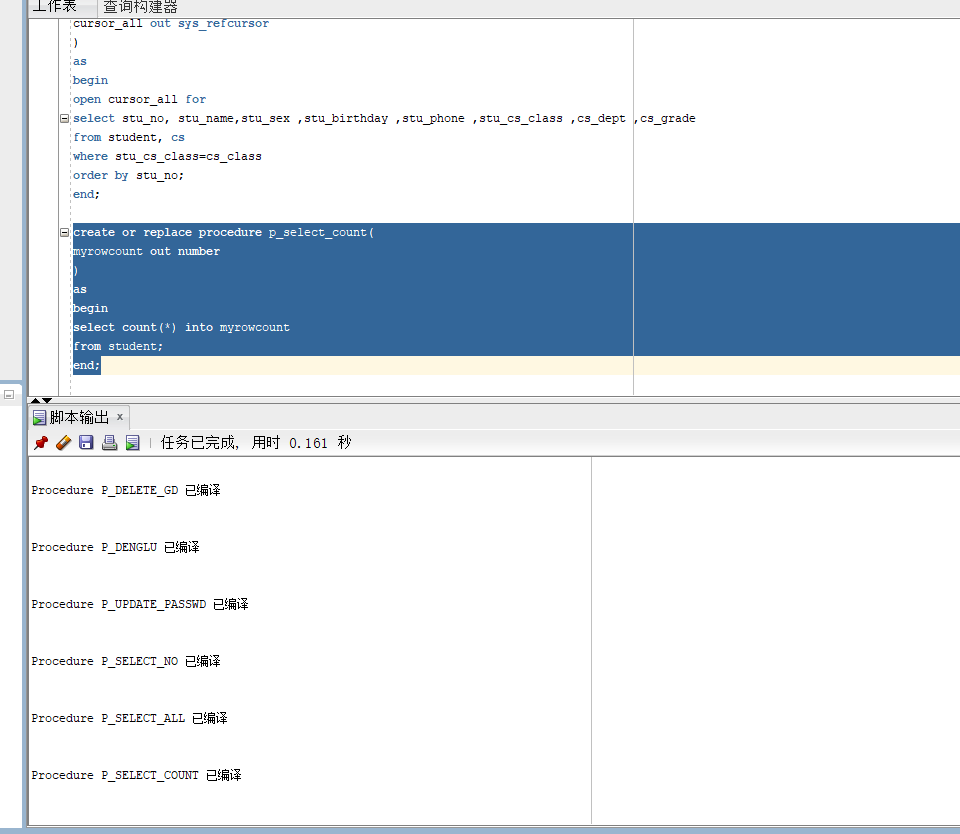
constraint fk\_gd\_sub\_subject foreign key(gd\_sub\_subject) references sub(sub\_subject)

);



4、创建存储过程

见第三章.源代码



5、插入数据

--插入数据

DECLARE

maxnumber CONSTANT NUMBER := 100;

i NUMBER := 1;

BEGIN

FOR i IN 1..maxnumber LOOP

INSERT INTO cs ( cs\_class,cs\_dept,cs\_grade ) VALUES (

concat('class\_',i),

concat('dept\_',i),

concat('grade\_',i)

);

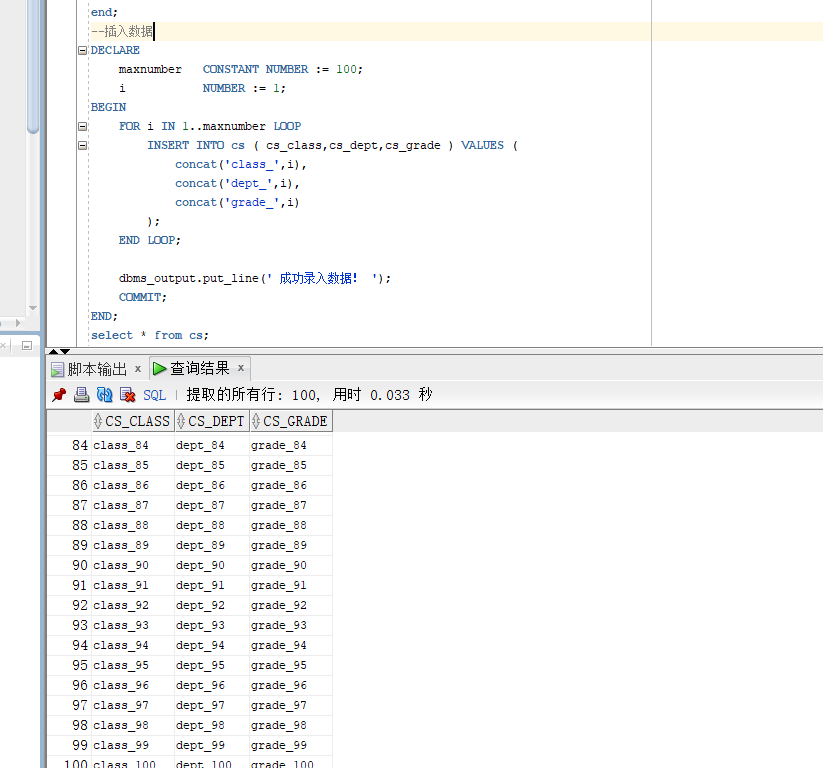
END LOOP;

dbms\_output.put\_line(' 成功录入数据！ ');

COMMIT;

END;

select \* from cs;



DECLARE

maxnumber CONSTANT NUMBER := 10000;

i NUMBER := 1;

teacher\_name NUMBER := 1;

BEGIN

FOR i IN 1..maxnumber LOOP

if mod(i,20)=0 then teacher\_name:=teacher\_name+1;

end if;

INSERT INTO sub ( sub\_subject,sub\_teacher\_name ) VALUES (

CONCAT('sub\_subject',i),

concat('tname',teacher\_name)

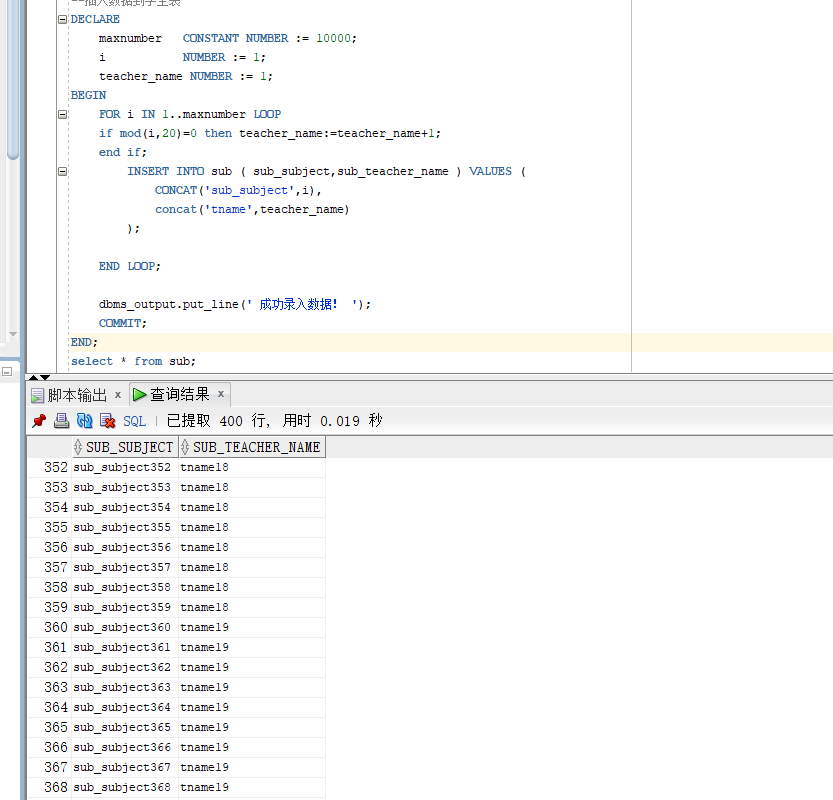
);

END LOOP;

dbms\_output.put\_line(' 成功录入数据！ ');

COMMIT;

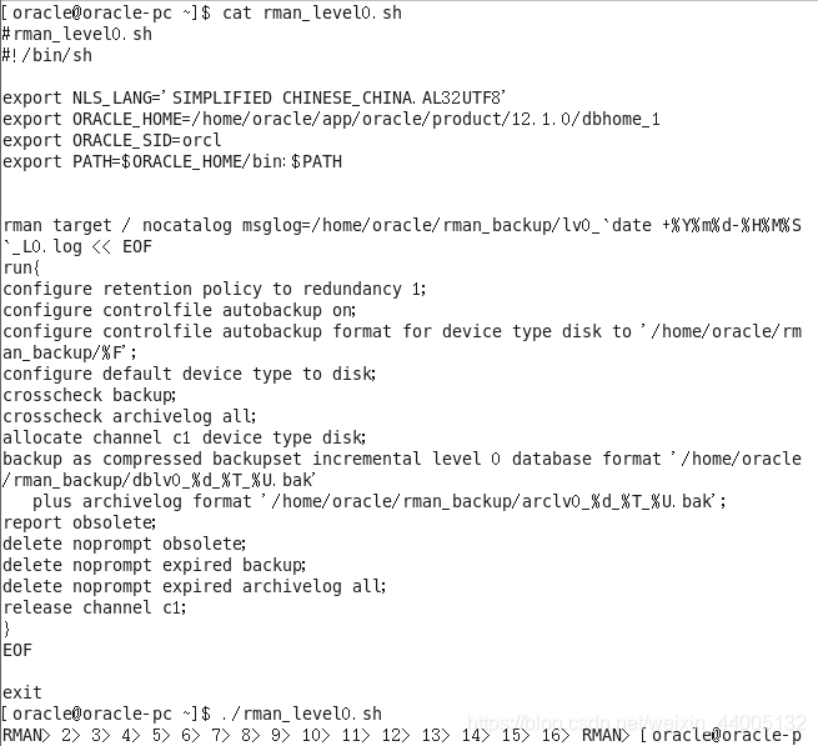
END;



6、全数据库备份

（1）开始备份

**$ cat rman\_level0.sh**



（2）查看备份文件内容

$ rman target /



（3）备份后修改数据

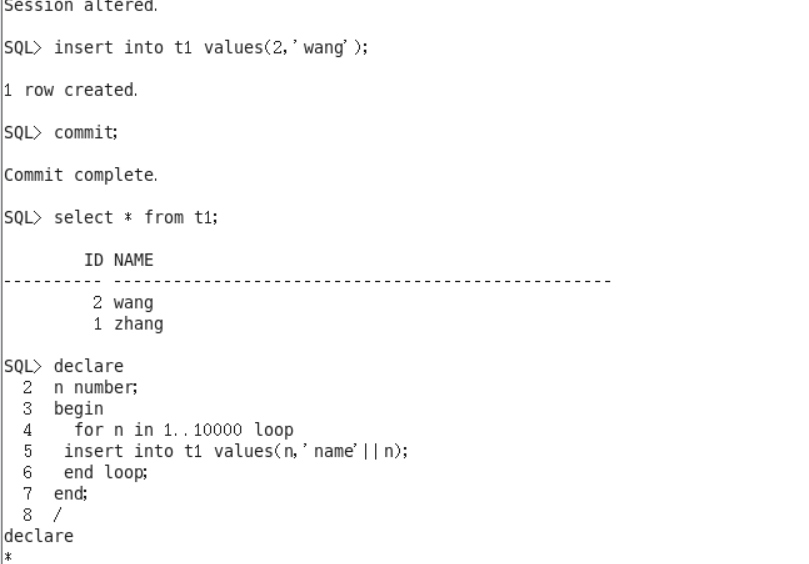


（4）删除数据库文件，模拟数据库文件损坏

**$ rm /home/oracle/app/oracle/oradata/orcl/pdborcl/pdbtest\_U123\_1.dbf**



（5）删除数据库文件后修改数据

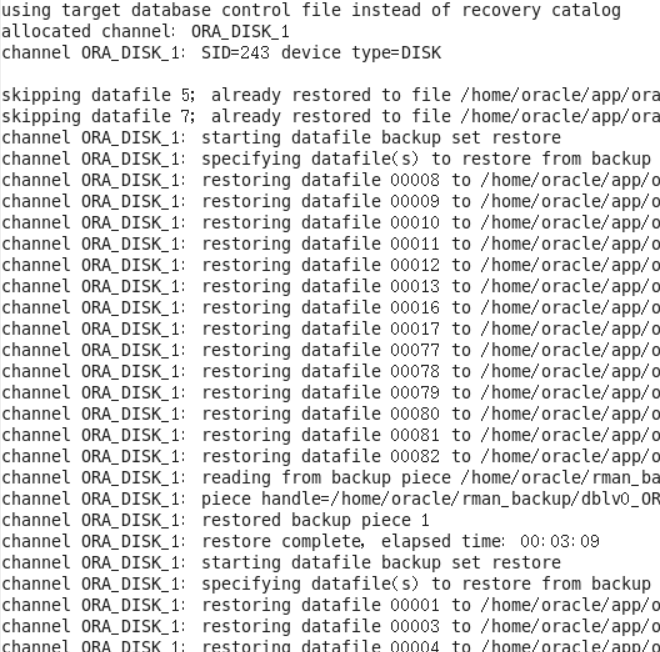


（6）数据库完全恢复

**重启损坏的数据库到mount状态**



**开始恢复数据库**

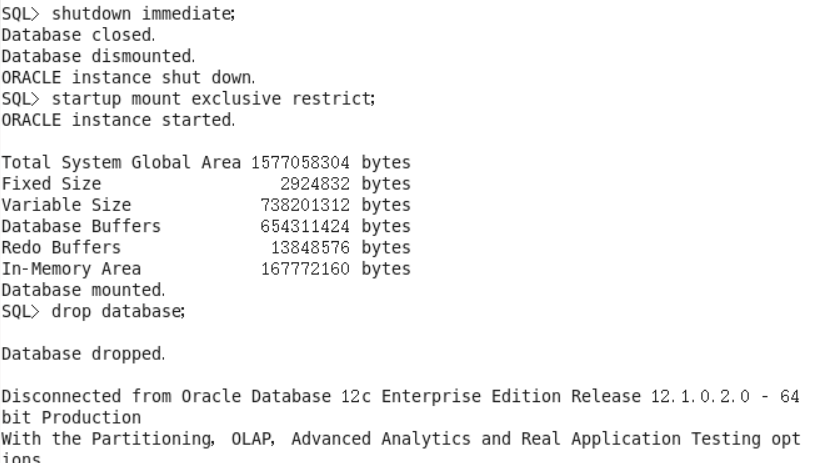


（7）设计容灾方法，使用两台主机，通过DataGuard实现数据库整体的异地备份

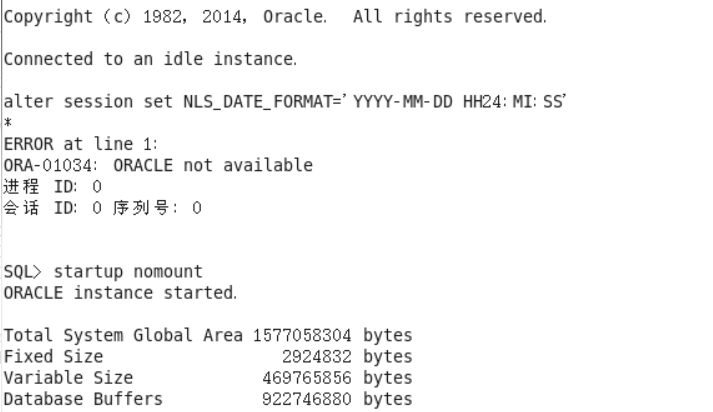
**备库**



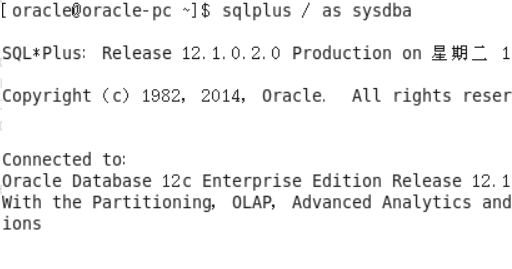
**删除原有数据库**



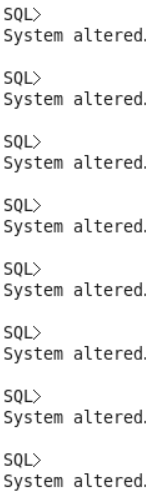
**启动到nomount7**



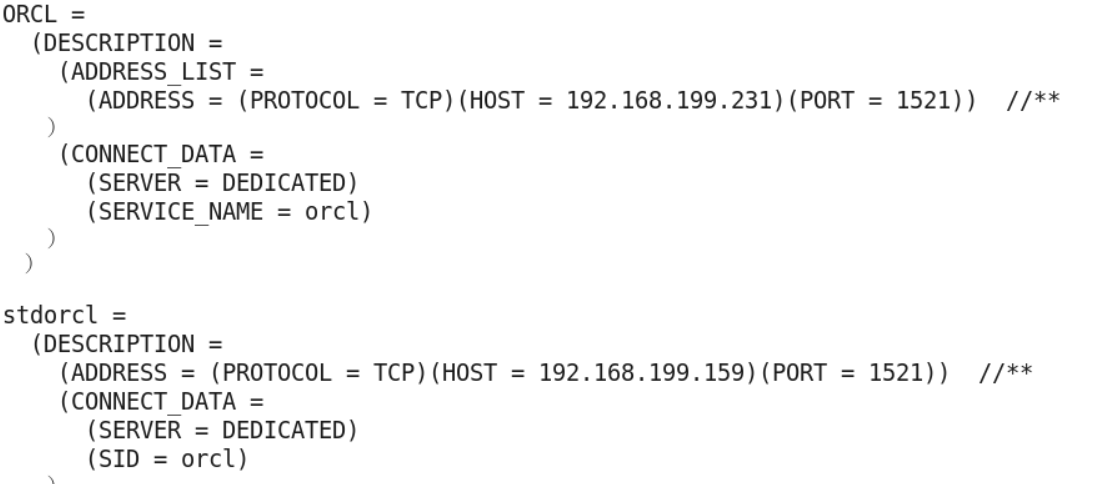
**主库**



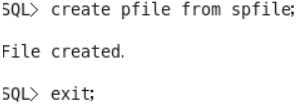
**主库环境开启强制归档**



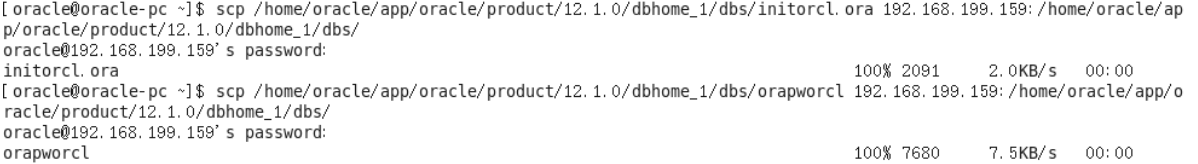
**编辑主库及备份的/home/oracle/app/oracle/product/12.1.0/dbhome\_1/network/admin/tnsnames.ora文件**



**生成/home/oracle/app/oracle/product/12.1.0/dbhome\_1/dbs/initorcl.ora**



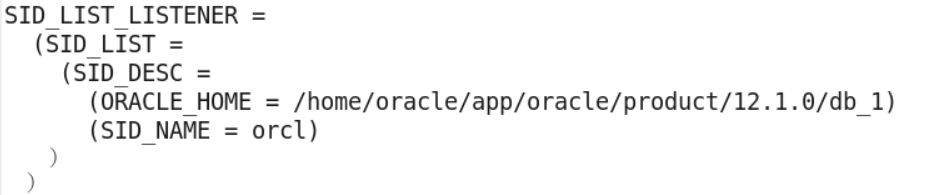
**将主库的参数文件，密码文件拷贝到备库**



**在备库上更改参数文件**



**在备库增加静态监听**



**重新启动，备库开启实时应用模式**

