

**期末项目设计报告**

|  |  |  |  |
| --- | --- | --- | --- |
| 题 目 | 基于Oracle的员工管理系统数据库设计 | | |
| 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 | 蒲镜名 | 学号 | 201810414317 |
| 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**

[1.项目简介 4](#_Toc74587604)

[2.功能分析 4](#_Toc74587605)

[2.1员工管理功能 4](#_Toc74587606)

[2.2部门管理功能 4](#_Toc74587607)

[2.3职位管理功能 5](#_Toc74587608)

[2.4项目管理功能 5](#_Toc74587609)

[2.5薪资管理功能 5](#_Toc74587610)

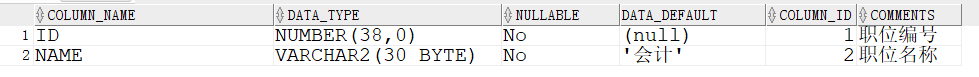
[3.项目设计 5](#_Toc74587611)

[4.数据库设计 5](#_Toc74587612)

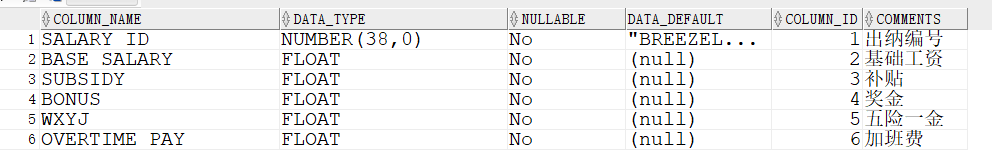
[4.1 EMPLOYEE\_INFO表（员工表） 6](#_Toc74587613)

[4.2 DEPARTMENTAL\_INFO表（部门表） 6](#_Toc74587614)

[4.3 DEPARTMENTAL\_POST表（职位表） 6](#_Toc74587615)

[4.4 PROJECT表（项目表） 6](#_Toc74587616)

[4.5 SALARY表（薪资表） 6](#_Toc74587617)

[5.设计项目涉及的表及表空间使用方案，并插入总共十万条数据 6](#_Toc74587618)

[5.1设计简介 6](#_Toc74587619)

[5.2创建表空间 7](#_Toc74587620)

[5.2.1代码部分 7](#_Toc74587621)

[5.2.2截图部分 8](#_Toc74587622)

[5.3创建表 8](#_Toc74587623)

[5.3.1代码部分 8](#_Toc74587624)

[5.3.2截图部分 12](#_Toc74587625)

[ 12](#_Toc74587626)

[5.4插入数据 12](#_Toc74587627)

[5.4.1存储过程设计 12](#_Toc74587628)

[5.4.2代码部分 12](#_Toc74587629)

[5.4.3截图部分 18](#_Toc74587630)

[6.设计权限及用户分配方案,两类角色，两个用户 20](#_Toc74587631)

[6.1设计简介 20](#_Toc74587632)

[6.2代码部分 20](#_Toc74587633)

[6.3截图部分 21](#_Toc74587634)

[7.在数据库中建立一个程序包，在包中用PL/SQL语言设计一些存储过程和函数，实现比较复杂的业务逻辑，用模拟数据进行执行计划分析 22](#_Toc74587635)

[7.1设计简介 22](#_Toc74587636)

[7.2代码与截图 23](#_Toc74587637)

[8.设计手动备份方案 24](#_Toc74587638)

[8.1设计简介 24](#_Toc74587639)

[8.2实验步骤（手动备份方案） 24](#_Toc74587640)

[8.2.1概念 24](#_Toc74587641)

[8.2.2步骤 24](#_Toc74587642)

[8.3实验步骤（联机自动备份） 25](#_Toc74587643)

[8.3.1概述 25](#_Toc74587644)

[8.3.2步骤 25](#_Toc74587645)

[8.3.3实际操作 26](#_Toc74587646)

[9.项目总结 28](#_Toc74587647)

# 1.项目简介

项目名称：员工管理系统

本项目是基于Oracle的员工管理系统数据库设计。

涉及角色/用户：管理员、普通用户

涉及表：员工表、部门表、职位表、项目表、薪资表

# 2.功能分析

## 2.1员工管理功能

员工管理功能，是由管理员进行的，主要包含员工查询、员工增加、员工修改、员工删除，主要通过操作数据库来实现这些操作。

## 2.2部门管理功能

部门管理功能，是由管理员进行的，主要包含部门查询、部门增加、部门修改、部门删除，主要通过操作数据库来实现这些操作。

## 2.3职位管理功能

职位管理功能主要指两个方面，主要包含职位管查询、职位管增加、职位管修改、职位管删除，主要通过操作数据库来实现这些操作。

## 2.4项目管理功能

项目管理能，是由管理员进行的，主要包含查看项目，修改项目起止时间、删除项目，创建项目等。主要通过操作数据库来实现这些操作。

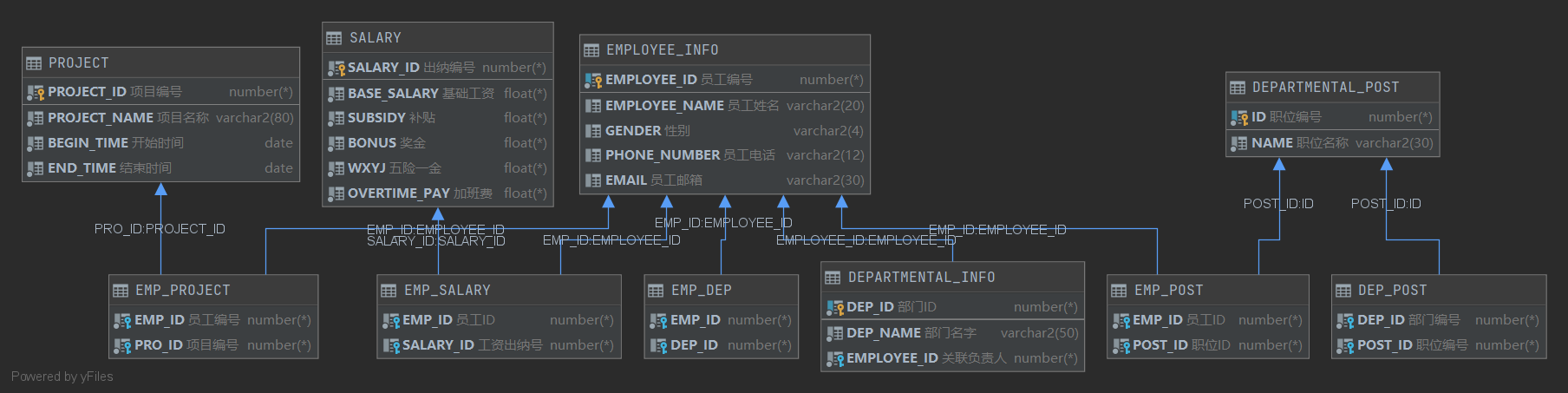
## 2.5薪资管理功能

薪资管理功能，是由管理员进行的，主要包含薪资查询、薪资结算、薪资修改，主要通过从数据库匹配关键信息来实现这些操作。

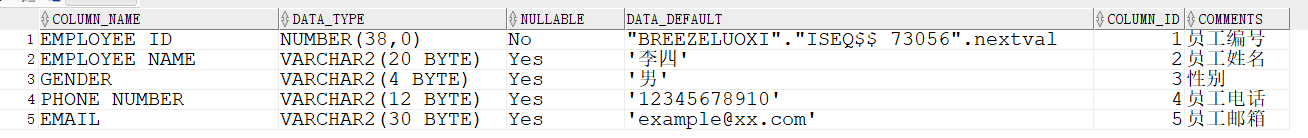
# 3.项目设计

项目主要包括数据结构设计，存储过程设计，备份恢复设计等。

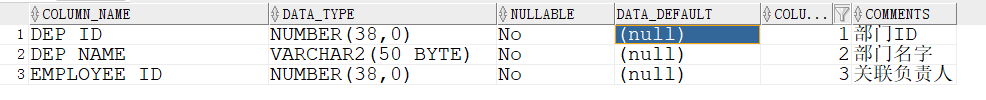
# 4.数据库设计



## 4.1 EMPLOYEE\_INFO表（员工表）

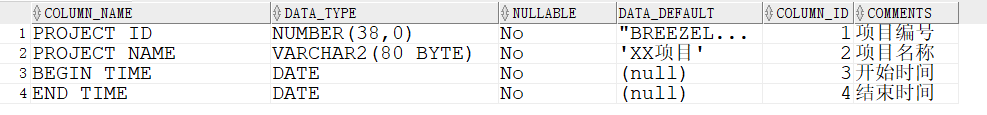


## 4.2 DEPARTMENTAL\_INFO表（部门表）



## 4.3 DEPARTMENTAL\_POST表（职位表）

## 4.4 PROJECT表（项目表）



## 4.5 SALARY表（薪资表）

## 5.设计项目涉及的表及表空间使用方案，并插入总共十万条数据

## 5.1设计简介

创建表空间：

表空间总共三个：breeze\_users，breeze\_users1

模型中包含五个实体，实体名称以及属性如下：

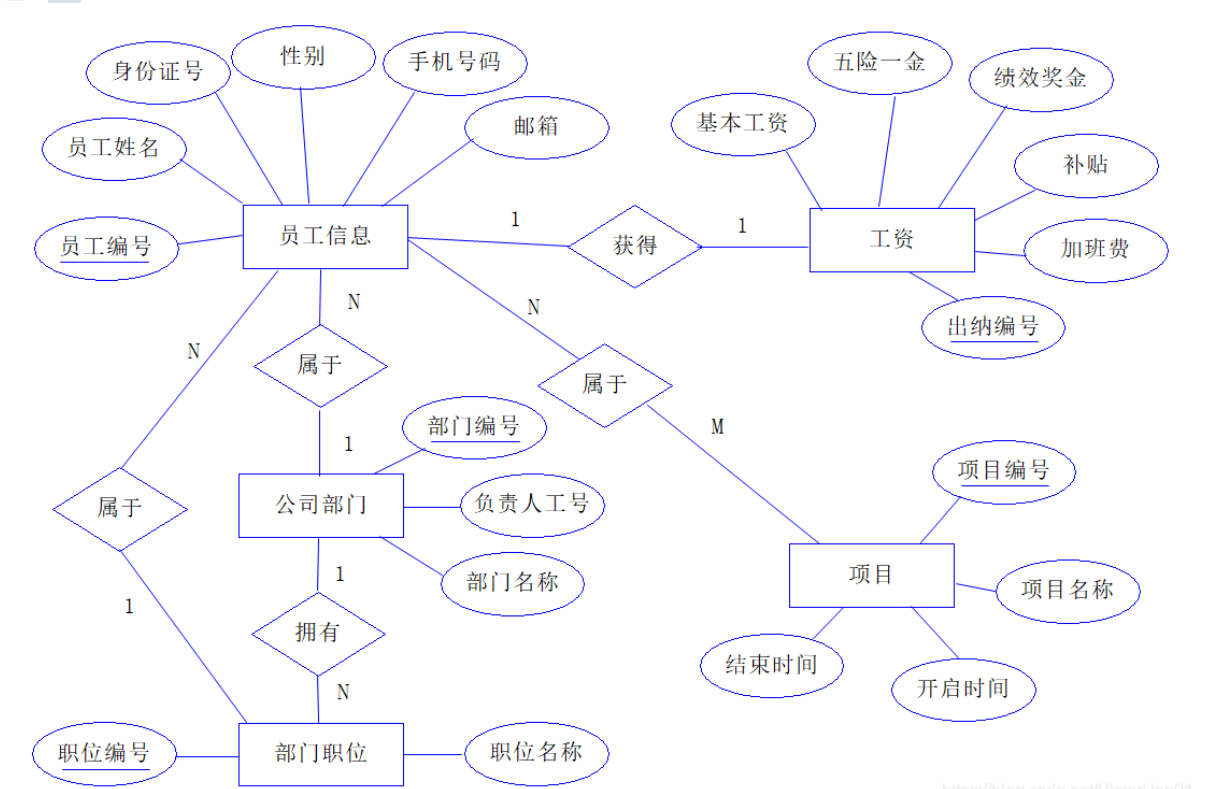
员工信息表：员工编号、员工姓名、身份证号、性别、手机号、邮箱；

公司部门表：部门编号、部门名称、负责人工号；

部门职位表：职位编号、职位名称；

项目表：项目编号、项目名称、开启时间、结束时间；

工资表：出纳编号、基本工资、五险一金、绩效奖金、补贴、加班费。



分区策略：

hash分区（散列）

散列分区为通过指定分区编号来均匀分布数据的一种分区类型，因为通过在I/O设备上进行散列分区，使得这些分区大小一致。如将物料交易表的数据根据交易ID散列地存放在指定的三个表空间中。

## 5.2创建表空间

### 5.2.1代码部分

-- 创建表空间breeze\_users  
CREATE TABLESPACE breeze\_users DATAFILE

'P:/Breeze/program/oradata/BREEZE/breeze\_users\_1.dbf'

SIZE 100M AUTOEXTEND ON NEXT 50M MAXSIZE UNLIMITED，

'P:/Breeze/program/oradata/BREEZE/breeze\_users\_2.dbf'

SIZE 100M AUTOEXTEND ON NEXT 50M MAXSIZE UNLIMITED

EXTENT MANAGEMENT LOCAL SEGMENT SPACE MANAGEMENT AUTO;

创建表空间breeze\_users1

CREATE TABLESPACE breeze\_users1 DATAFILE

'P:/Breeze/program/oradata/BREEZE/breeze\_users01\_1.dbf'

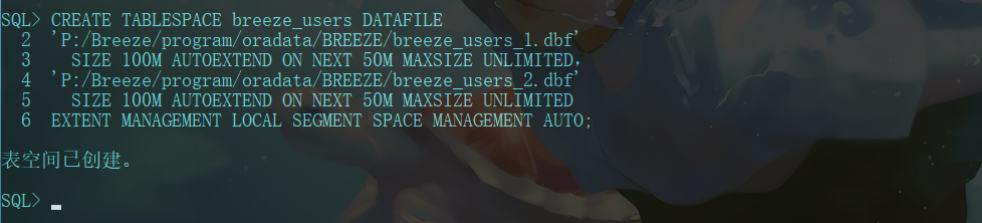
SIZE 100M AUTOEXTEND ON NEXT 50M MAXSIZE UNLIMITED，

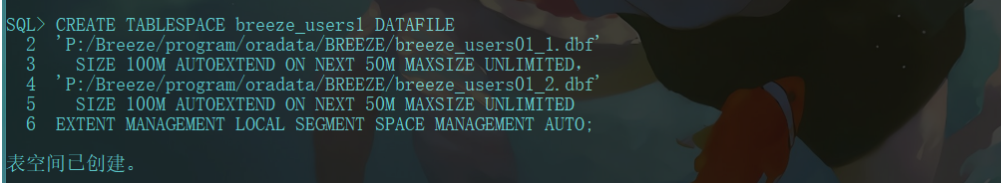
'P:/Breeze/program/oradata/BREEZE/breeze\_users01\_2.dbf'

SIZE 100M AUTOEXTEND ON NEXT 50M MAXSIZE UNLIMITED

EXTENT MANAGEMENT LOCAL SEGMENT SPACE MANAGEMENT AUTO;

### 5.2.2截图部分





## 5.3创建表

### 5.3.1代码部分

-- EMPLOYEE\_INFO表的创建

-- auto-generated definition

create table EMPLOYEE\_INFO

(

EMPLOYEE\_ID NUMBER default "BREEZELUOXI"."ISEQ$$\_73056".nextval generated as identity

constraint EMPLOYEE\_INFO\_PK

primary key,

EMPLOYEE\_NAME VARCHAR2(20) default '李四',

GENDER VARCHAR2(4) default '男',

PHONE\_NUMBER VARCHAR2(12) default '12345678910',

EMAIL VARCHAR2(30) default 'example@xx.com'

)

/

comment on table EMPLOYEE\_INFO is '员工信息表'

/

comment on column EMPLOYEE\_INFO.EMPLOYEE\_ID is '员工编号'

/

comment on column EMPLOYEE\_INFO.EMPLOYEE\_NAME is '员工姓名'

/

comment on column EMPLOYEE\_INFO.GENDER is '性别'

/

comment on column EMPLOYEE\_INFO.PHONE\_NUMBER is '员工电话'

/

comment on column EMPLOYEE\_INFO.EMAIL is '员工邮箱'

/

-- DEPARTMENTAL\_INFO 表创建

create table DEPARTMENTAL\_INFO

(

DEP\_ID NUMBER default "BREEZELUOXI"."ISEQ$$\_73059".nextval generated as identity

constraint DEPARTMENTAL\_INFO\_PK

primary key,

DEP\_NAME VARCHAR2(30) default 'XX部门',

EMPLOYEE\_ID NUMBER not null

constraint DEP\_EMPLOYEE

references EMPLOYEE\_INFO

)

/

comment on table DEPARTMENTAL\_INFO is '部门信息表'

/

comment on column DEPARTMENTAL\_INFO.DEP\_ID is '部门编号'

/

comment on column DEPARTMENTAL\_INFO.DEP\_NAME is '部门名称'

/

comment on column DEPARTMENTAL\_INFO.EMPLOYEE\_ID is '负责人工号'

/

-- DEPARTMENTAL\_POST 表创建

create table DEPARTMENTAL\_POST

(

ID NUMBER not null

constraint DEPARTMENTAL\_POST\_PK

primary key,

NAME VARCHAR2(30) default '会计' not null

)

/

comment on table DEPARTMENTAL\_POST is '部门职位'

/

comment on column DEPARTMENTAL\_POST.ID is '职位编号'

/

comment on column DEPARTMENTAL\_POST.NAME is '职位名称'

/

-- PROJECT 表创建

create table PROJECT

(

PROJECT\_ID NUMBER default "BREEZELUOXI"."ISEQ$$\_73067".nextval generated as identity

constraint PROJECT\_PK

primary key,

PROJECT\_NAME VARCHAR2(80) default 'XX项目' not null,

BEGIN\_TIME DATE not null,

END\_TIME DATE not null

)

/

comment on table PROJECT is '项目'

/

comment on column PROJECT.PROJECT\_ID is '项目编号'

/

comment on column PROJECT.PROJECT\_NAME is '项目名称'

/

comment on column PROJECT.BEGIN\_TIME is '开始时间'

/

comment on column PROJECT.END\_TIME is '结束时间'

/

-- SALARY 表创建

create table SALARY

(

SALARY\_ID NUMBER generated as identity

constraint SALARY\_PK

primary key,

BASE\_SALARY FLOAT not null,

SUBSIDY FLOAT not null,

BONUS FLOAT not null,

WXYJ FLOAT not null,

OVERTIME\_PAY FLOAT not null

)

/

comment on table SALARY is '工资'

/

comment on column SALARY.SALARY\_ID is '出纳编号'

/

comment on column SALARY.BASE\_SALARY is '基础工资'

/

comment on column SALARY.SUBSIDY is '补贴'

/

comment on column SALARY.BONUS is '奖金'

/

comment on column SALARY.WXYJ is '五险一金'

/

comment on column SALARY.OVERTIME\_PAY is '加班费'

/

### 5.3.2截图部分

## 

## 5.4插入数据

### 5.4.1存储过程设计

* SAVE\_TO\_DEPARTMENTAL\_POST,

存入部门职位表

* SAVE\_TO\_EMP\_DEP,

存入员工部门表

* SAVE\_TO\_EMP\_POST,

存入员工职位表

* SAVE\_TO\_EMP\_PROJECT,

存入员工项目表

* SAVE\_TO\_EMP\_SALARY,

存入员工薪资表

* SAVE\_TO\_EMPLOYEE\_INFO

存入员工表

* SQVE\_TO\_PROJECT,

存入工程项目总表

* SQVE\_TO\_SALARY

存入薪资总表

### 5.4.2代码部分

* SAVE\_TO\_DEPARTMENTAL\_POST,

create or replace PROCEDURE save\_to\_DEPARTMENTAL\_POST

AS

v\_name VARCHAR2(50);

BEGIN

for i in 1..20

loop

v\_name := '职位'||i||'号';

insert into DEPARTMENTAL\_POST (ID,NAME) VALUES (i,v\_name);

end loop;

END SAVE\_TO\_DEPARTMENTAL\_POST;

* SAVE\_TO\_EMP\_DEP,

create or replace procedure save\_to\_EMP\_DEP(

emp\_length in NUMBER

)

AS

--声明显式EMPLOYEE\_INFO表中游标

cursor c\_test is select EMPLOYEE\_ID from EMPLOYEE\_INFO;

-- 定义EMPLOYEE\_ID

v\_EMP\_id EMPLOYEE\_INFO.EMPLOYEE\_ID%type;

v\_index NUMBER;

begin DBMS\_OUTPUT.ENABLE(buffer\_size => null);

--打开游标

open c\_test;

-- 遍历游标

LOOP

FETCH c\_test into v\_EMP\_id;

-- 退出条件

exit when c\_test%NOTFOUND;

-- 生成随机index

SELECT trunc(dbms\_random.value(1,emp\_length+1))into v\_index FROM DUAL;

-- 打印信息

DBMS\_OUTPUT.PUT\_LINE('hello,emp\_id = '||v\_EMP\_id||' DEP\_ID = '|| v\_index);

-- 执行语句

insert into EMP\_DEP (EMP\_ID, DEP\_ID) values (v\_EMP\_id,v\_index);

end loop;

--关闭游标

close c\_test;

end save\_to\_EMP\_DEP;

* SAVE\_TO\_EMP\_POST,

create or replace PROCEDURE SAVE\_TO\_EMP\_POST(

post\_length in NUMBER

)

AS

cursor c\_test is select EMPLOYEE\_ID from EMPLOYEE\_INFO; --声明显式EMPLOYEE\_INFO表中游标

-- 定义EMPLOYEE\_ID

v\_EMP\_id EMPLOYEE\_INFO.EMPLOYEE\_ID%type;

v\_index NUMBER;

BEGIN DBMS\_OUTPUT.ENABLE(buffer\_size => null);

--打开游标

open c\_test;

-- 遍历游标

LOOP

FETCH c\_test into v\_EMP\_id;

-- 退出条件

exit when c\_test%NOTFOUND;

-- 生成随机index

SELECT trunc(dbms\_random.value(1,post\_length+1))into v\_index FROM DUAL;

-- 打印信息

DBMS\_OUTPUT.PUT\_LINE('hello,emp\_id = '||v\_EMP\_id||' POST\_ID = '|| v\_index);

-- 执行语句

insert into EMP\_POST (EMP\_ID, POST\_ID) values (v\_EMP\_id,v\_index);

end loop;

--关闭游标

close c\_test;

END SAVE\_TO\_EMP\_POST;

* SAVE\_TO\_EMP\_PROJECT,

create or replace PROCEDURE SAVE\_TO\_EMP\_PROJECT AS

-- 声明显式EMPLOYEE\_INFO表中游标

cursor c\_test is select EMPLOYEE\_ID from EMPLOYEE\_INFO;

-- 员工表中的ID

v\_EMP\_id EMPLOYEE\_INFO.EMPLOYEE\_ID%type;

-- 显示申明PROJECT 表中的游标

cursor c\_project is select PROJECT\_ID from PROJECT;

-- 项目表中的ID

v\_PROJECT\_id PROJECT.PROJECT\_ID%type;

BEGIN

-- 开启游标

open c\_test;

open c\_project;

-- 开启循环

loop

-- 取值

FETCH c\_test into v\_EMP\_id;

FETCH c\_project into v\_PROJECT\_id;

-- 退出

exit when c\_test%NOTFOUND or c\_project%NOTFOUND;

insert into EMP\_PROJECT (EMP\_ID,PRO\_ID) VALUES (v\_EMP\_id,v\_PROJECT\_id);

end loop;

-- 关闭游标

close c\_test;

close c\_project;

END SAVE\_TO\_EMP\_PROJECT;

* SAVE\_TO\_EMP\_SALARY,

create or replace PROCEDURE SAVE\_TO\_EMP\_SALARY(

month\_count IN number

)

AS

-- 声明显式EMPLOYEE\_INFO表中游标

cursor c\_test is select EMPLOYEE\_ID from EMPLOYEE\_INFO;

-- 声明显式SALARY表中游标

cursor c\_salary is select SALARY\_ID from SALARY;

-- 员工ID

v\_EMP\_id EMPLOYEE\_INFO.EMPLOYEE\_ID%type;

-- 工资单ID

v\_SALARY\_id SALARY.SALARY\_ID%type;

BEGIN

DBMS\_OUTPUT.ENABLE(buffer\_size => null);

--打开游标

open c\_test ;

open c\_salary;

-- 遍历员工ID

loop

FETCH c\_test into v\_EMP\_id;

-- 退出条件

exit when c\_test%NOTFOUND;

for i in 1..month\_count

loop

FETCH c\_salary into v\_SALARY\_id;

exit when c\_salary%NOTFOUND;

DBMS\_OUTPUT.PUT\_LINE('开始了empID = ' || v\_EMP\_id || 'POST\_ID = '|| v\_salary\_id);

insert into EMP\_SALARY (EMP\_ID,SALARY\_ID) VALUES (v\_EMP\_id,v\_salary\_id);

end loop;

end loop;

-- 关闭游标

close c\_test;

close c\_salary;

END SAVE\_TO\_EMP\_SALARY;

* SAVE\_TO\_EMPLOYEE\_INFO

create or replace PROCEDURE save\_to\_EMPLOYEE\_INFO

AS

v\_name varchar(50);

v\_gender varchar(4);

v\_phone varchar(12);

v\_email varchar(30);

begin

for i in 1..10000

loop

v\_name := '张三' || i || '号';

v\_gender := '男';

v\_phone := '135' || i;

v\_email := 'example@qq.com';

insert into BREEZELUOXI.EMPLOYEE\_INFO (EMPLOYEE\_NAME,GENDER,PHONE\_NUMBER,EMAIL) values (v\_name,v\_gender,v\_phone,v\_email);

end loop;

end;

* SQVE\_TO\_PROJECT,

create or replace PROCEDURE SQVE\_TO\_PROJECT AS

-- 声明显式EMPLOYEE\_INFO表中游标

cursor c\_test is select EMPLOYEE\_ID from EMPLOYEE\_INFO;

-- 员工ID

v\_EMP\_id EMPLOYEE\_INFO.EMPLOYEE\_ID%type;

-- 项目名称

v\_PROJECT\_NAME PROJECT.PROJECT\_NAME%type;

-- 项目开始时间

v\_BEGIN\_TIME PROJECT.BEGIN\_TIME%type;

-- 项目结束时间

v\_END\_TIME PROJECT.END\_TIME%type;

BEGIN

DBMS\_OUTPUT.ENABLE(buffer\_size => null);

-- 打开游标

open c\_test;

-- 循环遍历

loop

-- 取值

FETCH c\_test into v\_EMP\_id;

-- 是否退出

EXIT WHEN c\_test%NOTFOUND;

DBMS\_OUTPUT.PUT\_LINE('开始插入数据'||v\_EMP\_id||'号');

-- 插入数据操作

-- 初始化数据

v\_PROJECT\_NAME := '项目'||v\_EMP\_id||'号';

v\_BEGIN\_TIME := TO\_DATE(sysdate);

v\_END\_TIME := TO\_DATE(sysdate + 6000);

DBMS\_OUTPUT.PUT\_LINE('插入数据项目名 = '||v\_PROJECT\_NAME||'项目开始时间 = '||v\_begin\_time||'项目结束时间 = '|| v\_end\_time);

insert into PROJECT (PROJECT\_NAME,BEGIN\_TIME,END\_TIME)

VALUES (v\_PROJECT\_NAME,v\_BEGIN\_TIME,v\_END\_TIME);

end loop;

-- 关闭游标

close c\_test;

END SQVE\_TO\_PROJECT;

* SQVE\_TO\_SALARY

create or replace PROCEDURE SQVE\_TO\_SALARY(

month\_count in number

) AS

-- 声明显式EMPLOYEE\_INFO表中游标

cursor c\_test is select EMPLOYEE\_ID from EMPLOYEE\_INFO;

v\_EMP\_id number;

-- 基础工资

v\_BASE\_SALARY SALARY.BASE\_SALARY%type;

-- 补贴

v\_SUBSIDY SALARY.SUBSIDY%type;

-- 津贴

v\_BONUS SALARY.BONUS%type;

-- 五险一金

v\_WXYJ SALARY.WXYJ%type;

-- 加班工资

v\_OVERTIME\_PAY SALARY.OVERTIME\_PAY%type;

BEGIN

DBMS\_OUTPUT.ENABLE(buffer\_size => null);

--打开游标

open c\_test;

-- 遍历游标

LOOP

FETCH c\_test into v\_EMP\_id;

-- 退出条件

exit when c\_test%NOTFOUND;

DBMS\_OUTPUT.PUT\_LINE ('开始存储第'||v\_EMP\_id||'号员工的工资单');

for i in 1..month\_count

loop

v\_BASE\_SALARY := 6000.0 + i\*100.50; -- 6000底薪 + 月份\*100

v\_SUBSIDY := 600.85; -- 补贴

v\_BONUS := 800; -- 津贴

v\_WXYJ := i\*150.65; -- 五险一金

-- 生成随机加班工资

SELECT trunc(dbms\_random.value(10,500))into v\_OVERTIME\_PAY FROM DUAL;

INSERT into SALARY (BASE\_SALARY,SUBSIDY,BONUS,WXYJ,OVERTIME\_PAY) VALUES

(v\_BASE\_SALARY,v\_SUBSIDY,v\_BONUS,v\_WXYJ,v\_OVERTIME\_PAY);

DBMS\_OUTPUT.PUT\_LINE ('存入'||v\_BASE\_SALARY||v\_SUBSIDY||v\_BONUS||v\_WXYJ||v\_OVERTIME\_PAY);

end loop;

end loop;

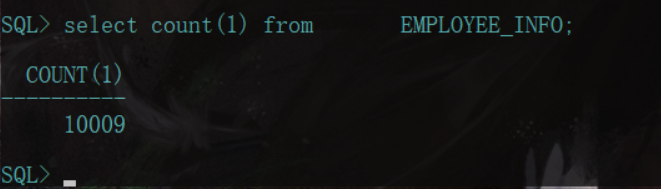
--关闭游标

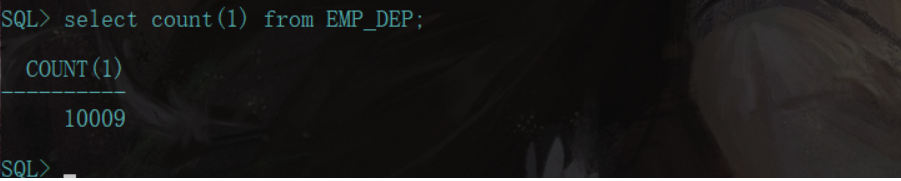
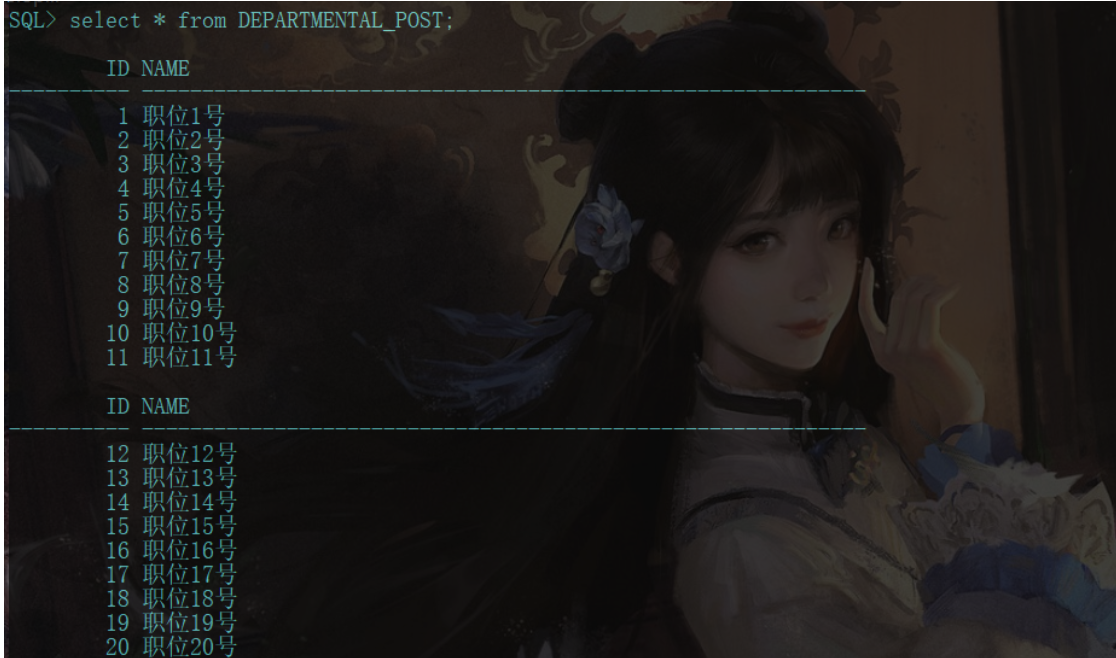
close c\_test;

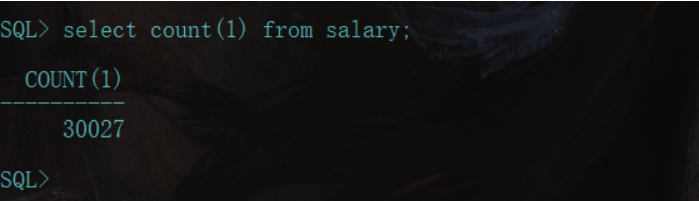
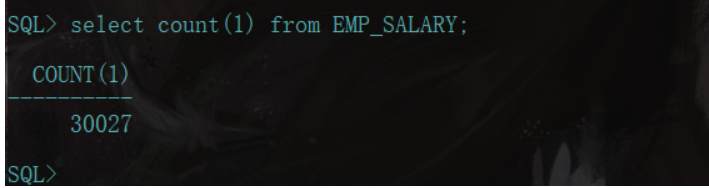
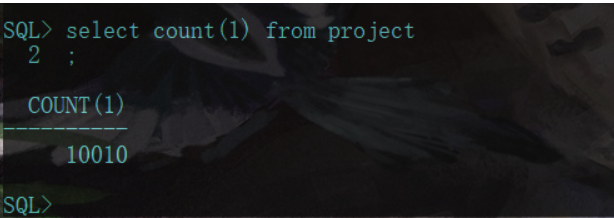
END SQVE\_TO\_SALARY;

### 5.4.3截图部分

验证数据是否插入成功：



# 6.设计权限及用户分配方案,两类角色，两个用户

## 6.1设计简介

用户：

Breezeluoxi

拥有角色：db\_manager

分配表空间：breeze\_users，breeze\_users1

Breezeluoxi1

拥有角色：db\_reader

分配表空间：breeze\_users，breeze\_users1

角色：

db\_manager

拥有角色：connect、resource、dba

db\_reader

拥有角色：connect、resource

## 6.2代码部分

-- 创建角色   
# 角色1

CREATE ROLE db\_manager IDENTIFIED BY pujingmin3344;

# 角色2

CREATE ROLE db\_reader IDENTIFIED BY 123456;  
-- 授权角色

grant create SESSION to db\_reader;

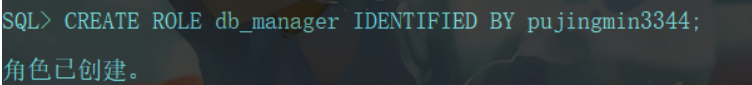
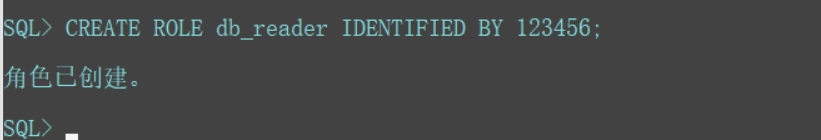
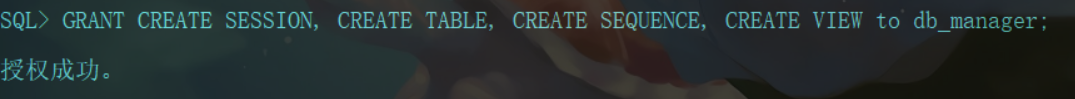
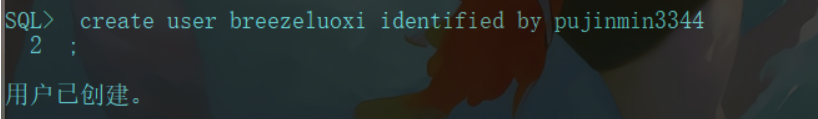
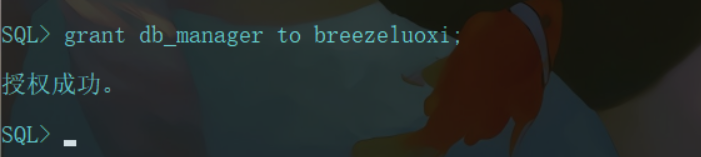
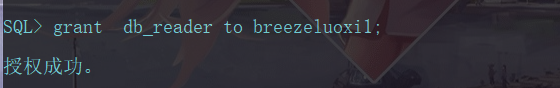
GRANT CREATE SESSION，CREATE TABLE,CREATE SEQUENCE，CREATE VIEW to db\_manager;  
  
-- 创建用户   
create user breezeluoxi identified by 123  
-- 指定用户额外表空间   
alter user breezeluoxi quota unlimited on breeze\_users;

alter user breezeluoxi quota unlimited on breeze\_users1;  
-- 创建用户   
create user breezeluoxi1 identified by 123456;

-- 指定用户额外表空间   
alter user breezeluoxi1 quota unlimited on breeze\_users;  
-- 分配角色给用户   
grant db\_manager to breezeluoxi;  
grant db\_reader to breezeluoxi1;

## 6.3截图部分

成功截图如下：

# 7.在数据库中建立一个程序包，在包中用PL/SQL语言设计一些存储过程和函数，实现比较复杂的业务逻辑，用模拟数据进行执行计划分析

## 7.1设计简介

建立的包：

包名：Work

函数：

函数名：get\_employee(employee\_id number)

功能：

输入员工ID，查询员工信息，所属部门，职位，薪水，参与项目

过程：

过程名：Get\_employee(employee\_id number)

功能：

输入员工ID，查询员工信息，所属部门，职位，薪水，参与项目

## 7.2代码与截图

-- 创建包  
create or replace PACKAGE TrainPack IS  
 FUNCTION Get\_count(order\_id\_t NUMBER) RETURN VARCHAR2;  
 PROCEDURE Get\_orders(train\_id\_t VARCHAR2);  
END TrainPack;  
-- 创建函数和过程  
create or replace PACKAGE BODY MY\_WORK IS

FUNCTION get\_employee(employee\_id number) RETURN VARCHAR2

AS

M VARCHAR2(100);

BEGIN

select EMPLOYEE\_NAME into N from EMPLOYEE\_INFO where EMPLOYEE\_ID=employee\_id;

RETURN N;

END;

PROCEDURE get\_employeeinfo(employee\_id number)

AS

v\_EMPLOYEE\_name employee\_info.employee\_name%type;

v\_DEPARTMENTAL\_name DEPARTMENTAL\_INFO.DEP\_NAME%type;

v\_DEP\_id emp\_dep.dep\_id%type;

v\_DEPARTMENTAL\_POST\_name departmental\_post.name%type;

v\_PROJECT\_name project.project\_name%TYPE;

v\_SALARY salary.base\_salary%TYPE;

begin

DBMS\_OUTPUT.ENABLE(buffer\_size => null);

--使用游标

DBMS\_OUTPUT.PUT\_LINE('输出员工信息：');

select EMPLOYEE\_NAME into v\_EMPLOYEE\_name from EMPLOYEE\_INFO where EMPLOYEE\_ID = employee\_id;

select DEP\_ID into v\_DEP\_id from EMP\_DEP where EMP\_DEP.EMP\_ID= employee\_id;

select DEP\_NAME into v\_DEPARTMENTAL\_name from DEPARTMENTAL\_INFO where DEP\_ID = v\_DEP\_id;

select name into v\_DEPARTMENTAL\_POST\_name from DEPARTMENTAL\_POST where ID = (select POST\_ID from EMP\_POST where EMP\_POST.EMP\_ID=employee\_id);

select project\_name into v\_PROJECT\_name from PROJECT where PROJECT\_ID = (select PRO\_ID from EMP\_PROJECT where EMP\_PROJECT.EMP\_ID=employee\_id);

select base\_salary into v\_SALARY from SALARY where SALARY\_ID = (select SALARY\_ID from EMP\_SALARY where EMP\_SALARY.EMP\_ID=employee\_id and rownum=1 );

DBMS\_OUTPUT.PUT\_LINE('员工姓名：'||v\_EMPLOYEE\_name||

'员工部门:'||v\_departmental\_name||

'员工职位：'||v\_departmental\_post\_name||

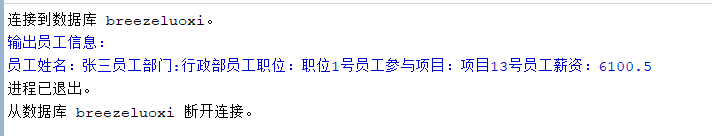
'员工参与项目：'||v\_project\_name||

'员工薪资：'||v\_salary);

END;

END TrainPack; /

测试：



# 8.设计手动备份方案

## 8.1设计简介

手动备份方案——脱机备份

自动备份方案——用户管理备份

## 8.2实验步骤（手动备份方案）

### 8.2.1概念

同脱机备份一样，用户管理备份也需要复制文件，但过程却完全不同

### 8.2.2步骤

* 查询所有的数据文件、控制文件和联机重做日志文件

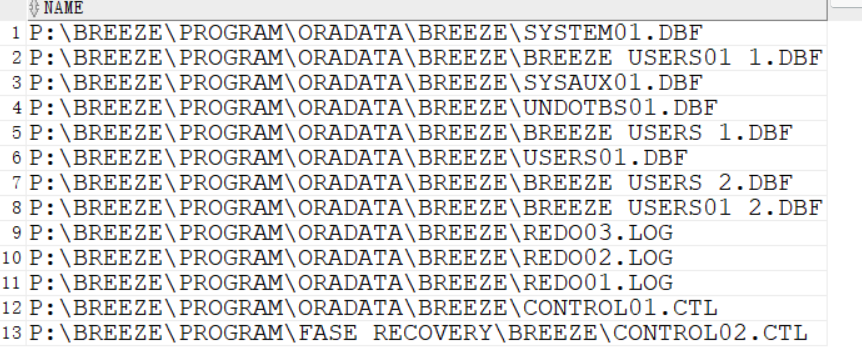
SELECT NAME FROM v$datafile

UNION ALL

SELECT MEMBER as NAME FROM v$logfile

UNION ALL

SELECT NAME FROM v$controlfile；



* 停机恢复

将手动备份的文件复制到原来的目录中重启数据库（startup）即可

### 8.3实验步骤（联机自动备份）

### 8.3.1概述

同脱机备份一样，用户管理备份也需要复制文件，但过程却完全不同

### 8.3.2步骤

1.让表空间进入备份模式

ALTER TABLESPACE \*\*\* BEGIN BACKUP

2.复制表空间文件

3.让表空间结束备份模式

ALTER TABLESPACE \*\*\* END BACKUP

4.复制控制文件和初始化文件

5.临时停止归档

ALTER SYSTEM SWITCH LOGFILE;

ALTER SYSTEM ARCHIVE LOG STOP;

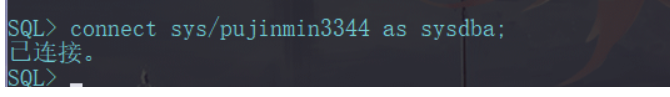
6.复制归档日志文件

7.重新开启归档

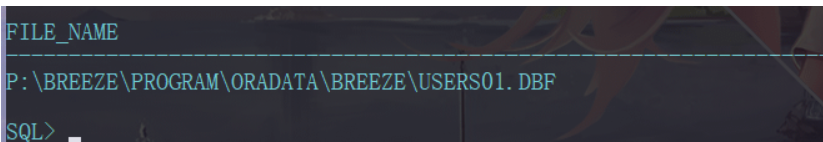
ALTER SYSTEM ARCHIVE LOG START;

### 8.3.3实际操作

* 以sys 用户登录

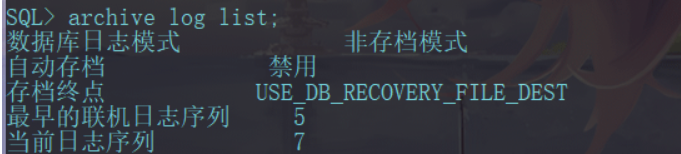


* 查询表空间的数据文件



* 进入备份模式

1. 查看归档模式archive log list;



1. 以archive模式启动数据库

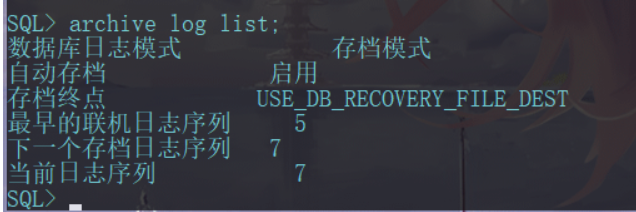
shutdown immediate

startup mount

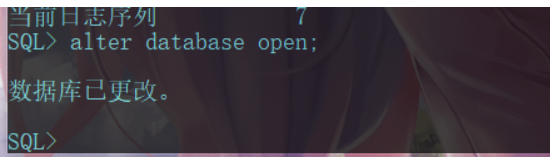
alter database archivelog;



1. 再次查看归档模式



1. alter database open;



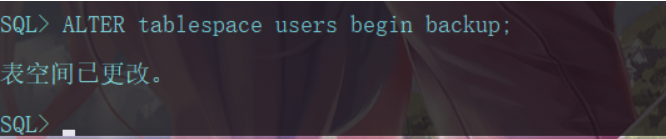
1. 转存数据库文件

!cp P:\BREEZE\PROGRAM\ORADATA\BREEZE\USERS01.DBF C:\Users\breez\Oracle\data\USERS01.DBF



1. 结束备份

ALTER tablespace users begin backup;



# 9.项目总结

完成本次期末项目我收获很多，对于Oracle数据库管理的方式，角色，对象，数据库他们的特性及意义有了自己的认识，明白了项目管理，数据库维护过程中对于用户的角色的限制是必要的，我们要做的就是在自己的职能范围之下完成自己的工作。

本次项目完成后，我对于oralce数据库操作有了一个全新的认识，oralce数据库的游标，存储过程，备份与恢复，等等功能，有了清晰的认识。收获最大的还是对于存储过程的学习，以及数据库备份功能的实现，明白了官方极力推崇存储过程的原因，另外对于数据备份与恢复我也有了自己的认识，备份与恢复是数据库管理中最重要的方面之一。如果数据库崩溃却没有办法恢复它，那么对企业造成的毁灭性结果可能会是数据丢失、收入减少、客户不满等。在任何情况下，无论数据库服务器有多稳定，可靠，我们都需要制订一个备份与恢复方案来备份重要数据并使自身免于灾难。

项目完成后我的一个解决问题的能力有了很大的提升也对于自己的不足有了清晰的认识，今后我会在好的地方继续发扬光大，不足的地方努力改正。