

**期末项目设计报告**

|  |  |  |  |
| --- | --- | --- | --- |
| 题 目 | 基于Oracle的成绩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 | 李林荫 | 学号 | 201810414312 |
| 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

[第1章 引言 4](#_Toc74529010)

[1.1编写目的 4](#_Toc74529011)

[1.2背景 4](#_Toc74529012)

[第2章 系统分析 5](#_Toc74529013)

[2.1需求分析 5](#_Toc74529014)

[2.2功能分析 5](#_Toc74529015)

[2.2.1 成绩管理模块 5](#_Toc74529016)

[2.2.2 课程管理模块 6](#_Toc74529017)

[2.2.3 班级管理模块 6](#_Toc74529018)

[2.2.4 学生管理模块 6](#_Toc74529019)

[2.2.5 专业管理模块 6](#_Toc74529020)

[2.2.6 教师管理模块 6](#_Toc74529021)

[2.2.7 系统管理模块 6](#_Toc74529022)

[第3章 数据库设计 7](#_Toc74529023)

[3.1概念结构设计 7](#_Toc74529024)

[3.3 物理结构设计 11](#_Toc74529025)

[第4章 表空间管理 14](#_Toc74529026)

[4.1表空间分析与设计 14](#_Toc74529027)

[4.2创建表空间 14](#_Toc74529028)

[4.2.1永久表空间的创建 14](#_Toc74529029)

[4.2.2临时表空间的创建 14](#_Toc74529030)

[4.2.3撤销表空间的创建 14](#_Toc74529031)

[4.3修改表空间 15](#_Toc74529032)

[第5章 表的使用 16](#_Toc74529033)

[5.1表的创建 16](#_Toc74529034)

[5.2索引 18](#_Toc74529035)

[5.3视图 18](#_Toc74529036)

[5.4使用序列 19](#_Toc74529037)

[5.4插入20000条数据 19](#_Toc74529038)

[第6章 查询 21](#_Toc74529039)

[6.1 SQL语言基础 21](#_Toc74529040)

[6.2 子查询与高级查询 21](#_Toc74529041)

[6.2.1查询平均成绩大于80分的同学的学号和平均成绩 21](#_Toc74529042)

[6.2.2查询“1”课程比“2”课程成绩高的所有学生的学号 21](#_Toc74529043)

[6.2.3查询姓“刘”的学生名单 21](#_Toc74529044)

[6.2.4查询1门以上不及格课程的同学的学号及其平均成绩 21](#_Toc74529045)

[第7章 PL/SQL语句 22](#_Toc74529046)

[7.1显示course表中的编号为1的课程名 22](#_Toc74529047)

[7.2 判断成绩61分所处的等级 22](#_Toc74529048)

[7.3判断学生的成绩是否及格 23](#_Toc74529049)

[第8章 存储过程 24](#_Toc74529050)

[8.1 创建一个存储过程update\_student 24](#_Toc74529051)

[8.2创建一个存储过程get\_grade\_information 24](#_Toc74529052)

[第9章 函数 26](#_Toc74529053)

[9.1 创建一个函数get\_sname 26](#_Toc74529054)

[9.2创建一个函数re\_stu\_info 26](#_Toc74529055)

[第10章 备份方案 28](#_Toc74529056)

[10.1 RMAN（备份与恢复管理器） 28](#_Toc74529057)

# 第1章 引言

## 1.1编写目的

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。

因此，开发设计这样一套学生成绩管理软件成为很有必要的事情。

## 1.2背景

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。

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需要一套底层数据库的支持。本文旨在对此系统所使用的数据库进行详细阐述。

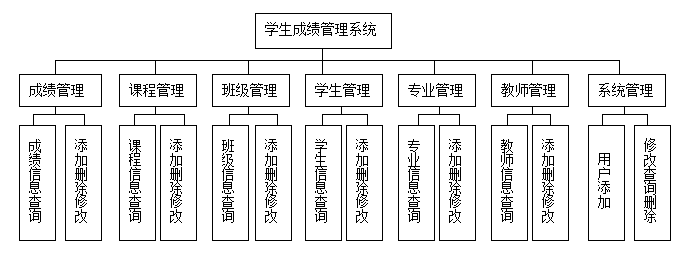
# 第2章 系统分析

## 2.1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

## 2.2功能分析

系统功能模块图如图1.1所示。



成绩信息查询

图1.1系统功能模块图

## 2.2.1 成绩管理模块

本模块实现有关成绩基本信息的录入、修改、查询与删除。同时能够实现按照课程代码或者学生学号进行查询并且当登陆身份为学生时，只能查询自己的成绩，当身份为教师时，可以录入学生成绩和查询成绩的功能，但是不可以修改和删除学生成绩。

## 2.2.2 课程管理模块

对各专业课程信息进行添加、删除、修改同时能够实现按照课程编号、课程名称进行查询。并且当登陆用户身份为学生或者任课教师时，只能进行查询课程信息操作。

## 2.2.3 班级管理模块

对班级信息进行添加、删除和修改，并且当登陆用户身份是学生或这任课教师时，只能进行查询班级信息的操作。

## 2.2.4 学生管理模块

有关学生基本信息的录入、修改查询和删除。同时能够实现按照学生的学号、姓名等条件进行查询，并且登陆用户身份是学生或者任课教师时只能进行学生信息查询。

## 2.2.5 专业管理模块

有关专业基本信息的录入、修改查询和删除。同时能够实现按照专业代码或专业名称进行查询，并且登陆身份是学生或者任课教师时，只能进行查询操作。

## 2.2.6 教师管理模块

有关教师基本信息的录入、修改查询和删除。同时能够实现按照教师号、姓名等信息进行查询，并且登陆用户为学生时不能对此模块进行任何操作。

## 2.2.7 系统管理模块

有系统管理员身份的用户可以登陆，负责用户管理。对用户进行添加修改、查询和删除功能。

# 第3章 数据库设计

## 3.1概念结构设计

列出系统E-R图，包括局部E-R图和总E-R图。

系统E-R图如图3.1所示：

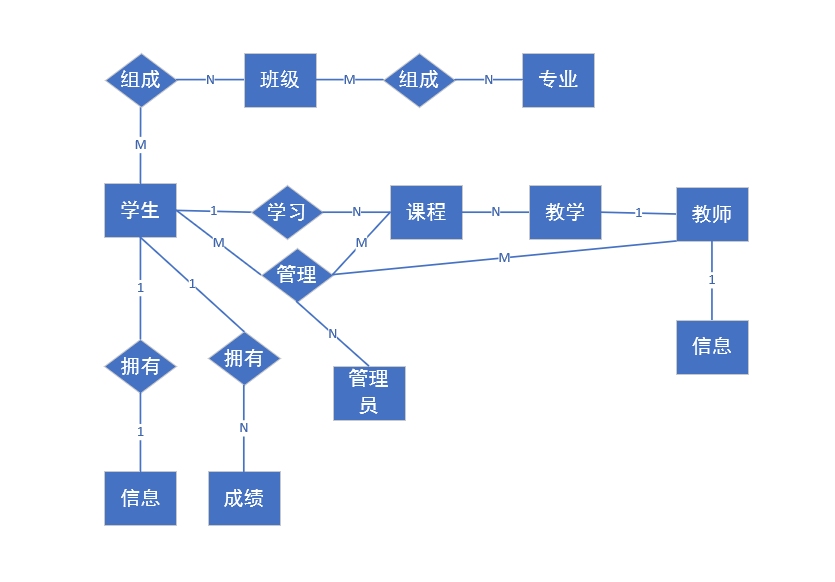


图3.1 学生成绩管理系统E-R图

学生基本信息实体图如图3.2所示：

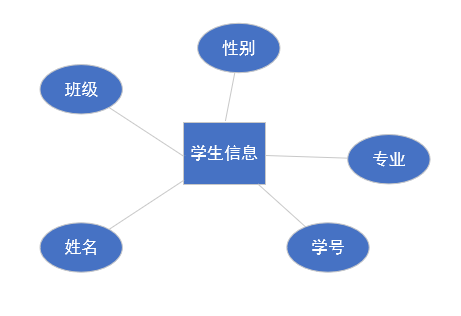


图3.2 学生基本信息实体图

教师基本信息实体图如图3.3所示：

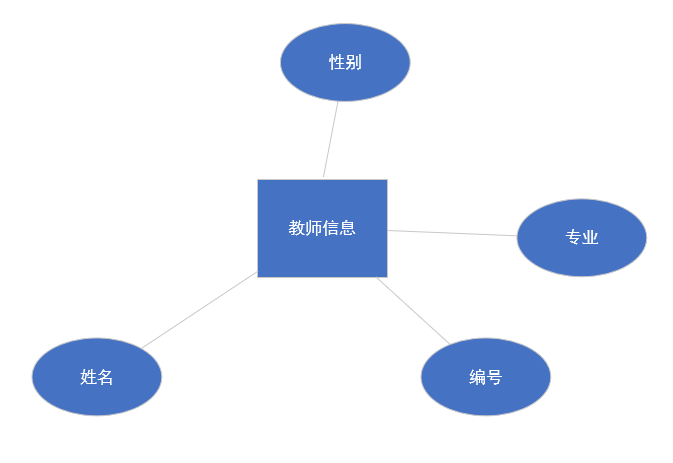


图3.3教师基本信息实体图

课程基本信息实体图如图3.4所示：

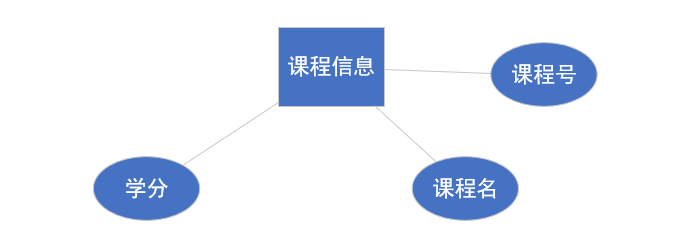


图3.4 课程基本信息实体图

专业信息实体图如图3.5所示：

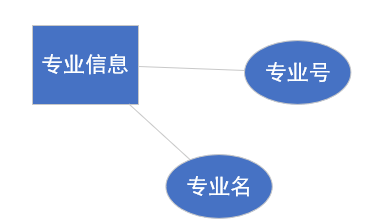


图3.5专业信息实体图

班级信息实体图如图3.6所示：

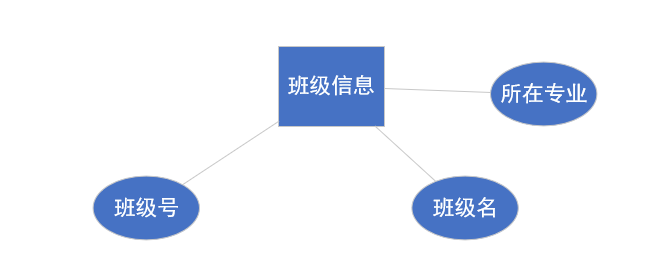


图3.6班级信息实体图

用户类别信息实体图如图3.7所示：

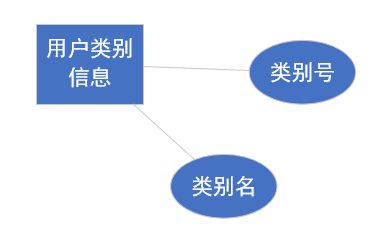


图3.7用户类别信息实体图

用户信息实体图如图3.8所示：

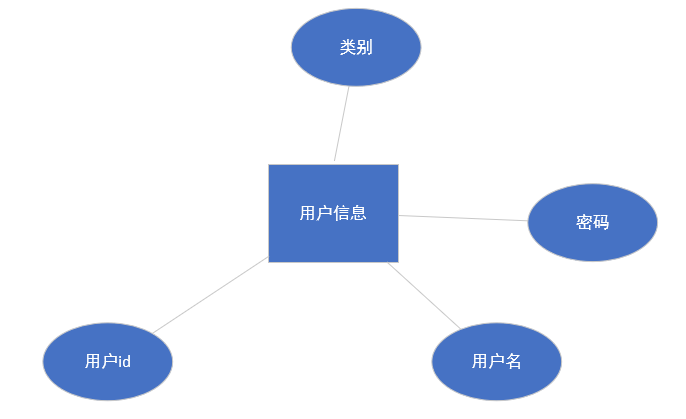


图3.8用户信息实体图

成绩信息实体图如图3.9所示：

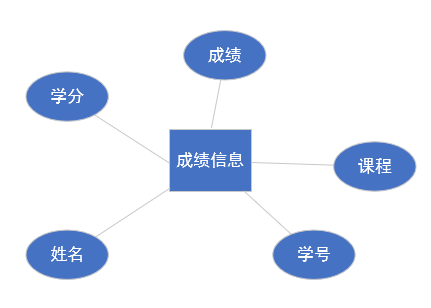


图3.9成绩信息实体图

## 3.3 物理结构设计

在数据库中创建名为studentspace的表空间，然后在表空间中根据关系模式创建表，其中表结构如下：

学生信息表，如表3.1所示。

表3.1 学生信息表(Student)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **字段名** | **数据类型** | **长度** | **是否为空** | **名称** |
| Sno | NUMBER | 10 | 否 | 学号 |
| Sname | VARCHAR2 | 4 | 否 | 学生姓名 |
| Sex | CHAR | 2 | 否 | 性别 |
| Classid | NUMBER | 10 | 否 | 班级号 |
| Majorid | NUMBER | 10 | 否 | 专业号 |

教师信息表，如表3.2所示。

表3.2 教师信息表(Teacher)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **字段名** | **数据类型** | **长度** | **是否为空** | **名称** |
| Tno | NUMBER | 10 | 否 | 教师编号 |
| Tname | VARCHAR2 | 4 | 否 | 教师姓名 |
| Sex | CHAR | 2 | 否 | 性别 |
| Majorid | NUMBER | 10 | 否 | 专业号 |

专业信息表，如表3.3所示。

表3.3 专业信息表(Major)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **字段名** | **数据类型** | **长度** | **是否为空** | **名称** |
| Majorid | NUMBER | 10 | 否 | 专业号 |
| Majorname | VARCHAR2 | 20 | 否 | 专业名 |

班级信息表，如表3.4所示。

表3.4 班级信息表(Class)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **字段名** | **数据类型** | **长度** | **是否为空** | **名称** |
| Classid | NUMBER | 10 | 否 | 班级号 |
| Classname | VARCHAR2 | 10 | 否 | 班级名 |
| Majorid | NUMBER | 10 | 否 | 所在专业 |

课程信息表，如表3.5所示。

表3.5 课程信息表(Course)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **字段名** | **数据类型** | **长度** | **是否为空** | **名称** |
| Cno | NUMBER | 10 | 否 | 课程编号 |
| Cname | VARCHAR2 | 10 | 否 | 课程名称 |
| Credit | NUMBER | 10 | 否 | 学分 |

成绩信息表，如表3.6所示。

表3.6 成绩信息表(Grade)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **字段名** | **数据类型** | **长度** | **是否为空** | **名称** |
| Sno | NUMBER | 10 | 否 | 学号 |
| Sname | VARCHAR2 | 10 | 否 | 学生姓名 |
| Cno | NUMBER | 10 | 否 | 课程编号 |
| Cname | VARCHAR2 | 20 | 否 | 课程名称 |
| Grade | NUMBER | 3 | 否 | 成绩 |
| Credit | NUMBER | 3 | 否 | 学分 |

用户信息表，如表3.7所示。

表3.7 用户信息表(Users)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **字段名** | **数据类型** | **长度** | **是否为空** | **名称** |
| Userid | VARCHAR2 | 10 | 否 | 登陆账号 |
| Uname | VARCHAR2 | 10 | 否 | 用户名 |
| Pwd | VARCHAR2 | 20 | 否 | 密码 |
| Typeid | NUMBER | 10 | 否 | 用户类别 |

用户类别信息表，如表3.8所示。

表3.8 用户类别信息表(Type)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **字段名** | **数据类型** | **长度** | **是否为空** | **名称** |
| Typeid | NUMBER | 10 | 否 | 类别编号 |
| Typename | VARCHAR2 | 10 | 否 | 类别名称 |

# 第4章 表空间管理

## 4.1表空间分析与设计

通过需求分析，首先为学生成绩管理系统创建永久表空间studentspace，存储学生成绩管理系统的数据信息。

其次为学生成绩管理系统创建临时表空间studenttemp，存储学生成绩管理系统中排序或汇总过程中产生的临时数据。

最后为学生成绩管理系统创建撤销表空间studentundo，存储学生成绩管理系统中修改前的数据，如果用户需要对数据进行恢复，就会使用到撤销表空间中存储的撤销数据。

## 4.2创建表空间

### 4.2.1永久表空间的创建

SQL> create tablespace studentspace

2 datafile 'D:\作业\oracle作业\shiyan\studentspace.dbf'

3 size 50m

4 autoextend on

5 next 5m

6 maxsize 100m;

### 4.2.2临时表空间的创建

SQL> create temporary tablespace studenttemp

2 tempfile 'D:\作业\oracle作业\shiyan\studenttemp.dbf'

3 size 10m

4 autoextend on

5 next 2m

6 maxsize 20m;

### 4.2.3撤销表空间的创建

SQL> create undo tablespace studentundo

2 datafile 'D:\作业\oracle作业\shiyan\studentundo.dbf'

3 size 50m

4 autoextend on

5 next 5m

6 maxsize 100m;

## 4.3修改表空间

4.3.1 通过数据字典dba\_data\_file查看studentspace表空间的数据文件信息

SQL> select tablespace\_name,file\_name,bytes

2 from dba\_data\_files

3 where tablespace\_name='STUDENTSPACE';

TABLESPACE\_NAME

------------------------------

FILE\_NAME

--------------------------------------------------

BYTES

----------

STUDENTSPACE

D:\作业\oracle作业\shiyan\STUDENTSPACE.DBF

52428800

4.3.2 修改studentspace表空间对应的数据文件的大小

SQL> alter database

2 datafile 'D:\作业\oracle作业\shiyan\STUDENTSPACE.DBF'

3 resize 40m;

4.3.3 为studentspace表空间添加一个新的数据文件

SQL> alter tablespace studentspace

2 add datafile

3 'D:\作业\oracle作业\shiyan\STUDENTSPACE1.DBF'

4 size 10m

5 autoextend on next 5m maxsize 40m;

4.3.4 删除新建的数据文件

SQL> alter tablespace studentspace

2 drop datafile 'D:\作业\oracle作业\shiyan\STUDENTSPACE1.DBF';

# 第5章 表的使用

## 5.1表的创建

5.1.1创建用户类别表，语句如下：

SQL> create table user\_type(

2 typeid number(10) primary key,

3 typename varchar2(10) not null

4 )tablespace studentspace;

表已创建。

5.1.2创建用户信息表，语句如下：

SQL> create table users(

2 userid varchar2(10) primary key,

3 uname varchar2(10) not null,

4 pwd varchar2(20) not null,

5 typeid number(10) not null,

6 constraint users\_type foreign key (typeid)

7 references type(typeid)

8 )tablespace studentspace;

表已创建。

5.1.3创建专业信息表，语句如下：

SQL> create table major(

2 majorid number(10) primary key,

3 majorname varchar(20) not null)

4 tablespace studentspace;

表已创建。

5.1.4创建班级信息表，语句如下：

SQL> create table class(

2 classid number(10) primary key,

3 classname varchar2(10) not null,

4 majorid number(10) not null,

5 constraint class\_major foreign key(majorid)

6 references major(majorid)

7 ) tablespace studentspace;

表已创建。

5.1.5创建学生信息表，语句如下：

SQL> create table student(

2 sno number(10) primary key,

3 sname varchar2(4) not null,

4 sex char(2) not null

5 check (sex in('男','女')),

6 classid number(10) not null,

7 majorid number(10) not null,

8 constraint student\_class foreign key(classid)

9 references class(classid),

10 constraint student\_major foreign key(majorid)

11 references major(majorid)

12 )tablespace studentspace;

表已创建。

5.1.6创建教师信息表，语句如下：

SQL> create table teacher(

2 tno number(10) primary key,

3 tname varchar2(4) not null,

4 sex char(2) not null

5 check (sex in('男','女')),

6 majorid number(10) not null,

7 constraint teacher\_major foreign key(majorid)references major(majorid)

8 )tablespace studentspace;

表已创建。

5.1.7创建课程信息表，语句如下：

SQL> create table course(

2 cno number(10) primary key,

3 cname varchar(20) unique not null,

4 credit number(2) not null)tablespace studentspace;

表已创建。

5.1.8创建学生成绩表，语句如下：

SQL> create table grade(

2 sno number(10) primary key,

3 sname varchar2(10) not null,

4 cno number(10) not null,

5 cname varchar2(20) not null,

6 grade number(3) not null,

7 credit number(3) not null,

8 constraint grade\_student foreign key(sno)references student(sno),

9 constraint grade\_course foreign key(cno)references course(cno)

10 )tablespace studentspace;

表已创建。

## 5.2索引

5.2.1在Class表中的classname列上创建classname\_index的索引

SQL> create index classname\_index

2 on class(classname)

3 tablespace studentspace;

5.2.2 打开Class表中class列上的classid\_index索引的监控状态

SQL> alter index classname\_index monitoring usage;

索引已更改。

通过数字字典v$object\_usage可以查看哪些索引正在被监控

SQL> column index\_name format a15;

SQL> column table\_name format a15;

SQL> select index\_name,table\_name,monitoring,

2 used,start\_monitoring,end\_monitoring

3 from v$object\_usage;

INDEX\_NAME TABLE\_NAME MON USE START\_MONITORING END\_MONITORING

--------------- --------------- --- --- ------------------- ----------------

CLASSNAME\_INDEX CLASS YES NO 05/21/2016 11:00:43

## 5.3视图

5.3.1创建基于class表和magor表的视图V1，在该视图的子查询中检索班级信息的同时显示其所在专业名称。

SQL> create view v1

2 as

3 select c.classid,c.classname,m.majorname

4 from class c left join major m

5 on c.majorid=m.majorid;

视图已创建。

5.3.2创建基于grade表的视图V2,查询成绩不及格学生的信息

SQL> create view v2

2 as

3 select sno,sname,cname,grade from grade where grade<60;

视图已创建。

## 5.4使用序列

创建一个名为student\_seq的序列

SQL> create sequence student\_seq

2 start with 1

3 increment by 1

4 nocache

5 nocycle

6 order;

序列已创建。

## 5.4插入20000条数据

对表插入数据。

```

-- student表数据插入20000

declare

i int;

sno number(20);

sname VARCHAR2(100);

sex VARCHAR2(100);

classid number(20);

majorid number(20);

begin

i:=1;

while i<=20000

loop

sno:=i;

sname:= ''|| i;

sex := '男'|| i;

classid := '123'|| i;

majorid := '123'|| i;

insert into train\_( sno,sname,sex,classid,majorid) values (sno,sname,sex,classid,majorid);

i:=i+1;

end loop;

commit;

end;

/

-- grade表数据插入20000

declare

i int;

sno number(20);

sname VARCHAR2(100);

cno VARCHAR2(100);

cname number(20);

grade number(20);

credit number(3)

begin

i:=1;

while i<=20000

loop

sno:=i;

sname:= ''|| i;

sex := '男'|| i;

classid := '123'|| i;

majorid := '123'|| i;

insert into train\_( sno,sname,cno,cname,grade,credit) values( sno,sname,cno,cname,grade,credit);

i:=i+1;

end loop;

commit;

end;

/

# 第6章 查询

## 6.1 SQL语言基础

统计各专业女同学人数

SQL> select majorid,count(\*) from student

2 where sex='女'

3 group by majorid

4 having count(\*)>0

5 order by count(\*) desc;

## 6.2 子查询与高级查询

### 6.2.1查询平均成绩大于80分的同学的学号和平均成绩

SQL> select sno, avg(grade) from grade

2 group by sno

3 having avg(grade)>80;

### 6.2.2查询“1”课程比“2”课程成绩高的所有学生的学号

SQL> select a.sno

2 from (select \* from grade s where s.cno = 1) a,

3 (select \* from grade s where s.cno = 2) b

4 where a.sno = b.sno

5 and a.grade > b.grade;

### 6.2.3查询姓“刘”的学生名单

SQL> select \* from student where sname like '刘%';

### 6.2.4查询1门以上不及格课程的同学的学号及其平均成绩

SQL> select sno, avg(grade) from grade

2 group by sno having sno in(

3 select sno from grade

4 where grade<60

5 group by sno

6 having count(\*) >1);

# 第7章 PL/SQL语句

## 7.1显示course表中的编号为1的课程名

使用PL/SQL程序块，输出显示course表中的编号为1的课程名。

SQL> set serveroutput on;

SQL> declare

2 id constant number(10):=1;

3 name varchar2(30);

4 begin

5 select cname into name

6 from course where cno=id;

7 dbms\_output.put\_line(id||name);

8 end;

9 /

1oracle

PL/SQL 过程已成功完成。

## 7.2 判断成绩61分所处的等级

在PL/SQL中，使用if条件语句判断成绩61分所处的等级。

SQL> declare

2 score binary\_integer:=61;

3 begin

4 if score >=90 then

5 dbms\_output.put\_line('优秀');

6 elsif score>=80 then

7 dbms\_output.put\_line('良好');

8 elsif score>=70 then

9 dbms\_output.put\_line('中等');

10 elsif score>=60 then

11 dbms\_output.put\_line('及格');

12 else

13 dbms\_output.put\_line('不及格');

14 end if;

15 end;

16 /

及格

## 7.3判断学生的成绩是否及格

在PL/SQL中，查询所有学生的成绩是否有不及格，如有不及格就触发异常并输出。

SQL> declare

2 cursor c1 is select sname from grade where grade<60;

3 one grade.sname%type;

4 e1 exception;

5 begin

6 open c1;

7 fetch c1 into one;

8 if c1%found then raise e1;

9 end if;

10 exception

11 when e1 then

12 dbms\_output.put\_line(one||'不达标');

13 close c1;

14 end;

15 /

杨不达标

PL/SQL 过程已成功完成。

# 第8章 存储过程

## 8.1 创建一个存储过程update\_student

创建一个存储过程update\_student，该过程用来将student表中的编号为1的学生的姓名改为”一库”。

SQL> create procedure update\_student

2 as

3 begin

4 update student set sname='一库' where sno =1;

5 end update\_student;

6 /

过程已创建。

使用execute语句调用存储过程，如下：

SQL> execute update\_student;

PL/SQL 过程已成功完成。

## 8.2创建一个存储过程get\_grade\_information

采取直接在存储过程中使用DBMS\_OUTPUT.PUT\_LINE过程输出相关内容。

SQL> create or replace procedure get\_grade\_information

2 (s\_no number)

3 as

4 s\_name varchar2(10);

5 c\_no number;

6 c\_name varchar2(20);

7 s\_grade number(3);

8 c\_credit number(3);

9 begin

10 select sname,cno,cname,grade,credit

11 into s\_name,c\_no,c\_name,s\_grade,c\_credit

12 from grade where sno=s\_no;

13 DBMS\_OUTPUT.PUT\_LINE('学生姓名：'||s\_name);

14 DBMS\_OUTPUT.PUT\_LINE('课程编号：'||c\_no);

15 DBMS\_OUTPUT.PUT\_LINE('课程名称：'||c\_name);

16 DBMS\_OUTPUT.PUT\_LINE('分数：'||s\_grade);

17 DBMS\_OUTPUT.PUT\_LINE('学分：'||c\_credit);

18 end get\_grade\_information;

19 /

过程已创建。

调用get\_grade\_information存储过程。例如获取sno为1401的学生的成绩信息，如下：

SQL> set serveroutput on

SQL> exec get\_grade\_information(1402);

学生姓名：杨

课程编号：2

课程名称：java

分数：40

学分：8

PL/SQL 过程已成功完成。

# 第9章 函数

## 9.1 创建一个函数get\_sname

创建一个函数get\_sname，该函数实现按sno获取sname，函数创建如下：

SQL> create function get\_sname(stu\_num number)

2 return varchar2 as

3 stu\_name student.sname%type;

4 begin

5 select sname into stu\_name from student where sno=stu\_num;

6 return stu\_name;

7 end get\_sname;

8 /

函数已创建。

因为函数是具有返回值的，所以它类似于一个表达式，调用函数可以直接使用select语句，如下：

SQL> select get\_sname(1) from dual;

GET\_SNAME(1)

-----------------------------------------------------------------

shen

## 9.2创建一个函数re\_stu\_info

创建一个函数re\_stu\_info,以班级号为参数，返回各个班级总平均分。

SQL> create or replace function re\_stu\_info(cl\_id number) return varchar2 is

2 v\_result varchar2(100);

3 cursor cur\_class is

4 select a.classname as classname,

5 round(avg(grade), 2) as avg\_score

6 from class a

7 inner join student b

8 on a.classid = b.classid

9 inner join grade c

10 on c.sno = b.sno

11 where a.classid = cl\_id

12 group by a.classname;

13 c\_row cur\_class%rowtype;

14 begin

15 for c\_row in cur\_class loop

16 dbms\_output.put\_line('班级：' || c\_row.classname ||

17 '总平均分为：' || c\_row.avg\_score);

18 end loop;

19 return v\_result;

20 end;

21 /

调用函数：

SQL> declare

2 t varchar2(50);

3 v\_number number(10);

4 cursor cur is select classid from class;

5 begin

6 open cur;

7 fetch cur into v\_number;

8 while cur%found loop

9 t:=re\_stu\_info(v\_number);

10 fetch cur into v\_number;

11 end loop;

12 close cur;

13 end;

14 /

# 第10章 备份方案

## 10.1 RMAN（备份与恢复管理器）

1.为目录创建一个单独的表空间

SQL>Create tablespace tools datafile ‘fielname’ size 50m;

2.创建RMAN用户

SQL>Create user RMAN identified by RMAN default tablespace tools temporary tablespace temp;

3.给RMAN授予权限

SQL>Grant connect , resource , recovery\_catalog\_owner to rman;

4.打开RMAN

$>RMAN

5.连接数据库

RMAN>connect catalog rman/rman

6.创建恢复目录

RMAN>Create catalog tablespace tools

注册目标数据库，恢复目录创建成功后，就可以注册目标数据库了，目标数据库就是需要备份的数据库，一个恢复目录可以注册多个目标数据库，注册目标数据库的命令为：

$>RMAN target internal/password catalog rman/rman@rcdb;

RMAN>Register database;

数据库注册完成,就可以用RMAN来进行备份了，更多命令请参考ORACLE联机手册或《ORACLE8i备份与恢复手册》。

注销数据库不是简单的在RMAN提示下反注册就可以了，需要运行一个程序包，过程如下：

（1）. 连接目标数据库，获得目标数据库ID

$> RMAN target internal/password catalog rman/rman@rcdb;

RMAN-06005: connected to target database: RMAN (DBID=1231209694)

（2）. 查询恢复目录，得到更详细的信息

SQL> SELECT db\_key, db\_id FROM db WHERE db\_id = 1231209694;

DB\_KEY DB\_ID

---------- ---------------

1 1237603294

1 row selected.

（3）.运行过程dbms\_rcvcat.unregisterdatabase注销数据库，如

SQL> EXECUTE dbms\_rcvcat.unregisterdatabase(1 , 1237603294)

（4）采用RMAN进行备份

1).备份整个数据库

backup full tag ‘basicdb’ format ‘/bak/oradata/full\_%u\_%s\_%p’ database;

2).备份一个表空间

backup tag ‘tsuser’ format ‘/bak/oradata/tsuser\_%u\_%s\_%p’ tablespace users;

3).备份归档日志

backup tag ‘alog’ format ‘/bak/archivebak/arcbak\_%u\_%s\_%p’ archivelog all delete input;