

**期末项目设计报告**

|  |  |  |  |
| --- | --- | --- | --- |
| 题 目 | 基于Oracle的医患管理数据库设计 | | |
| 专 业 | 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 | 龙可凡 | 学号hao号 | 201810414303 |
| 指导老师 | 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 | 权限及用户元配方案设计正确 | 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

# 医患管理系统

目录

[医患管理系统 2](#_Toc74614346)

[引言 3](#_Toc74614347)

[1 表设计 3](#_Toc74614348)

[1.1 病患信息表 3](#_Toc74614349)

[1.2 医生信息表 3](#_Toc74614350)

[1.3 类别信息表 4](#_Toc74614351)

[1.4 科室信息表 4](#_Toc74614352)

[1.5 项目信息表 4](#_Toc74614353)

[1.6 诊断结果信息表 4](#_Toc74614354)

[1.7 用户信息表 5](#_Toc74614355)

[1.8 用户类别信息表 5](#_Toc74614356)

[2 创建表空间 5](#_Toc74614357)

[2.1 永久表空间的创建 5](#_Toc74614358)

[2.2 临时表空间的创建 5](#_Toc74614359)

[2.3 撤销表空间的创建 6](#_Toc74614360)

[3 修改表空间 6](#_Toc74614361)

[3.1 通过数据字典查看表空间 6](#_Toc74614362)

[3.2 修改表空间对应的数据文件的大小 6](#_Toc74614363)

[3.3 为表空间添加一个新的数据文件 6](#_Toc74614364)

[3.4 删除新建的数据文件 7](#_Toc74614365)

[4 表的相关操作 7](#_Toc74614366)

[4.1 表的创建 7](#_Toc74614367)

[4.2 索引 9](#_Toc74614368)

[4.3 视图 10](#_Toc74614369)

[4.4 使用序列 10](#_Toc74614370)

[5 查询 12](#_Toc74614371)

[5.1 SQL语言基础 12](#_Toc74614372)

[5.2 子查询与高级查询 12](#_Toc74614373)

[6 SQL语句 13](#_Toc74614374)

[6.1 显示表中的编号为1的类别名 13](#_Toc74614375)

[6.2 判断诊断结果61分所处的等级 14](#_Toc74614376)

[6.3 判断患者的诊断结果是否健康 14](#_Toc74614377)

[7 创建存储过程 15](#_Toc74614378)

[7.1 创建一个存储过程update\_patient 15](#_Toc74614379)

[7.2 创建一个存储过程get\_Result\_information 15](#_Toc74614380)

[8 函数设计 16](#_Toc74614381)

[8.1 创建一个函数get\_Pname 16](#_Toc74614382)

[8.2 创建一个函数re\_pat\_info 17](#_Toc74614383)

[9 备份方案 18](#_Toc74614384)

[9.1 RMAN（备份与恢复管理器） 18](#_Toc74614385)

## 引言

医患管理系统是一个医院必不可少的一部分，随着互联网的普及和计算机的普遍使用，医患管理系统，有了更大的发展空间，通过对医患管理系统的开发可以提高医务人员以及患者就诊的工作效率，因此开发设计这样一套医患管理系统软件是很必要的事情。

医患管理系统是一个医院不可缺少的部分，一个良好的医患管理系统，可以方便用户进行预约挂号，病例查询等等的操作，医生在系统上对病人发布他的就诊诊断信息，对医生查看患者的过往病逝以及诊断病人的病情有十分重要的作用。由于各个医院在这方面都有需求，所以如何管理，这些庞大的数据就变得更加的复杂，传统的医患管理系统，不仅工作量大，而且很容易出现问题，有效率低，保密性差的缺点。

但是目前我们通过技术手段可以很好的解决这些问题，借助sql来对医生和患者的数据今进行管理，使医院的管理更上一层楼，方便大众，服务大众。

## 1 表设计

### 1.1 病患信息表

表1.1 病患信息表(Patient)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **字段名** | **数据类型** | **长度** | **是否为空** | **名称** |
| Pno | NUMBER | 10 | 否 | 病号 |
| Pname | VARCHAR2 | 4 | 否 | 病患姓名 |
| Sex | CHAR | 2 | 否 | 性别 |
| Cateid | NUMBER | 10 | 否 | 类别号 |
| Departid | NUMBER | 10 | 否 | 科室号 |

### 1.2 医生信息表

表1.2 医生信息表(Doctor)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **字段名** | **数据类型** | **长度** | **是否为空** | **名称** |
| Dno | NUMBER | 10 | 否 | 医生编号 |
| Dname | VARCHAR2 | 4 | 否 | 医生姓名 |
| Sex | CHAR | 2 | 否 | 性别 |
| Departid | NUMBER | 10 | 否 | 科室号 |

### 1.3 类别信息表

表1.3 类别信息表(Cate)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **字段名** | **数据类型** |  | **长度** | **是否为空** | **名称** |
| Cateid | NUMBER |  | 10 | 否 | 科室号 |
| DepPname | VARCHAR2 |  | 10 | 否 | 科室名 |
| Departid | NUMBER |  | 10 | 否 | 所在科室 |

### 1.4 科室信息表

表1.4 科室信息表(Depart)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **字段名** | **数据类型** | **长度** | **是否为空** | **名称** |
| Departid | NUMBER | 10 | 否 | 科室号 |
| DeparDname | VARCHAR2 | 20 | 否 | 科室名 |

### 1.5 项目信息表

表1.5 项目信息表(Project)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **字段名** | **数据类型** | **长度** | **是否为空** | **名称** |
| Cno | NUMBER | 10 | 否 | 项目编号 |
| Cname | VARCHAR2 | 10 | 否 | 项目名称 |
| Credit | NUMBER | 10 | 否 | 指标 |

### 1.6 诊断结果信息表

表1.6 诊断结果信息表(Result)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **字段名** | **数据类型** | **长度** | **是否为空** | **名称** |
| Pno | NUMBER | 10 | 否 | 病号 |
| Pname | VARCHAR2 | 10 | 否 | 病患姓名 |
| Cno | NUMBER | 10 | 否 | 项目编号 |
| Cname | VARCHAR2 | 20 | 否 | 项目名称 |
| Result | NUMBER | 3 | 否 | 诊断结果 |
| Credit | NUMBER | 3 | 否 | 指标 |

### 1.7 用户信息表

表1.7 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 |

### 1.8 用户类别信息表

表1.8 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 |

## 2 创建表空间

### 2.1 永久表空间的创建

SQL> create tablespace hosspace

2 datafile 'D:\homework\oraclehomework\BoBo\hosspace.dbf'

3 size 50m

4 autoextend on

5 next 5m

6 maxsize 100m;

### 2.2 临时表空间的创建

SQL> create temporary tablespace patienttemp

2 tempfile 'D:\homework\oraclehomework\BoBo\patienttemp.dbf'

3 size 10m

4 autoextend on

5 next 2m

6 maxsize 20m;

### 2.3 撤销表空间的创建

SQL> create undo tablespace hosundo

2 datafile 'D:\homework\oraclehomework\BoBo\hosundo.dbf'

3 size 50m

4 autoextend on

5 next 5m

6 maxsize 100m;

## 3 修改表空间

### 3.1 通过数据字典查看表空间

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

2 from dba\_data\_files

3 where tablespace\_name='HOSSPACE';

TABLESPACE\_NAME

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

FILE\_NAME

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

BYTES

----------

HOSSPACE

D:\homework\oraclehomework\BoBo\HOSSPACE.DBF

52428800

### 3.2 修改表空间对应的数据文件的大小

SQL> alter database

2 datafile 'D:\homework\oraclehomework\BoBo\HOSSPACE.DBF'

3 resize 40m;

### 3.3 为表空间添加一个新的数据文件

SQL> alter tablespace hosspace

2 add datafile

3 'D:\homework\oraclehomework\BoBo\HOSSPACE1.DBF'

4 size 10m

5 autoextend on next 5m maxsize 40m;

### 3.4 删除新建的数据文件

SQL> alter tablespace hosspace

2 drop datafile 'D:\homework\oraclehomework\BoBo\HOSSPACE1.DBF';

## 4 表的相关操作

### 4.1 表的创建

#### 4.1.1 创建用户类别表

SQL> create table user\_type(

2 typeid number(10) primary key,

3 typename varchar2(10) not null

4 )tablespace patientspace;

表已创建。

#### 4.1.2 创建用户信息表

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 patientspace;

表已创建。

#### 4.1.3 创建科室信息表

SQL> create table Depart(

2 Departid number(10) primary key,

3 DeparDname varchar(20) not null)

4 tablespace patientspace;

表已创建。

#### 4.1.4 创建类别信息表

SQL> create table Cate(

2 Cateid number(10) primary key,

3 DepPname varchar2(10) not null,

4 Departid number(10) not null,

5 constraint Cate\_Depart foreign key(Departid)

6 references Depart(Departid)

7 ) tablespace patientspace;

表已创建。

#### 4.1.5 创建病患信息表

SQL> create table patient(

2 Pno number(10) primary key,

3 Pname varchar2(4) not null,

4 sex char(2) not null

5 check (sex in('男','女')),

6 Cateid number(10) not null,

7 Departid number(10) not null,

8 constraint patient\_Cate foreign key(Cateid)

9 references Cate(Cateid),

10 constraint patient\_Depart foreign key(Departid)

11 references Depart(Departid)

12 )tablespace patientspace;

表已创建。

#### 4.1.6 创建医生信息表

SQL> create table doctor(

2 Dno number(10) primary key,

3 Dname varchar2(4) not null,

4 sex char(2) not null

5 check (sex in('男','女')),

6 Departid number(10) not null,

7 constraint doctor\_Depart foreign key(Departid)references Depart(Departid)

8 )tablespace patientspace;

表已创建。

#### 4.1.7 创建项目信息表

SQL> create table project(

2 cno number(10) primary key,

3 cname varchar(20) unique not null,

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

表已创建。

#### 4.1.8 创建诊断结果信息表

SQL> create table Result(

2 Pno number(10) primary key,

3 Pname varchar2(10) not null,

4 cno number(10) not null,

5 cname varchar2(20) not null,

6 Result number(3) not null,

7 credit number(3) not null,

8 constraint Result\_patient foreign key(Pno)references patient(Pno),

9 constraint Result\_project foreign key(cno)references project(cno)

10 )tablespace patientspace;

表已创建。

### 4.2 索引

#### 4.2.1 在表中的列上创建索引

SQL> create index DepPname\_index

2 on Cate(DepPname)

3 tablespace patientspace;

#### 4.2.2 打开表中列上的索引的监控状态

SQL> alter index DepPname\_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

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

DEPPNAME\_INDEX CATE YES NO 06/04/2021 15:04:26

### 4.3 视图

#### 4.3.1 创建基于Cate表和Depart表的视图

SQL> create view v1

2 as

3 select c.Cateid,c.DepPname,m.DeparDname

4 from Cate c left join Depart m

5 on c.Departid=m.Departid;

视图已创建。

#### 4.3.2 创建基于Result表的视图

SQL> create view v2

2 as

3 select Pno,Pname,cname,Result from Result where Result<60;

视图已创建。

### 4.4 使用序列

创建一个名为patient\_seq的序列

SQL> create sequence patient\_seq

2 start with 1

3 increment by 1

4 nocache

5 nocycle

6 order;

序列已创建。

对表插入数据。

```

-- patient表数据插入20000

declare

i int;

Pno number(20);

Pname VARCHAR2(100);

sex VARCHAR2(100);

Cateid number(20);

Departid number(20);

begin

i:=1;

while i<=20000

loop

Pno:=i;

Pname:= ''|| i;

sex := '男'|| i;

Cateid := '123'|| i;

Departid := '123'|| i;

insert into train\_( Pno,Pname,sex,Cateid,Departid) values (Pno,Pname,sex,Cateid,Departid);

i:=i+1;

end loop;

commit;

end;

/

-- Result表数据插入20000

declare

i int;

Pno number(20);

Pname VARCHAR2(100);

cno VARCHAR2(100);

cname number(20);

Result number(20);

credit number(3)

begin

i:=1;

while i<=20000

loop

Pno:=i;

Pname:= ''|| i;

sex := '男'|| i;

Cateid := '123'|| i;

Departid := '123'|| i;

insert into train\_( Pno,Pname,cno,cname,Result,credit) values( Pno,Pname,cno,cname,Result,credit);

i:=i+1;

end loop;

commit;

end;

/

## 5 查询

### 5.1 SQL语言基础

统计各专业女同学人数

SQL> select Departid,count(\*) from patient

2 where sex='女'

3 group by Departid

4 having count(\*)>0

5 order by count(\*) desc;

### 5.2 子查询与高级查询

#### 5.2.1 查询诊断结果评分大于80分的病患的编号和平均诊断结果

SQL> select Pno, avg(Result) from Result

2 group by Pno

3 having avg(Result)>80;

#### 5.2.2 查询“1”诊断结果比“2“诊断结果评分高的所有病患编号

SQL> select a.Pno

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

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

4 where a.Pno = b.Pno

5 and a.Result > b.Result;

#### 5.2.3 查询姓“刘“的病患名单

SQL> select \* from patient where Pname like '刘%';

#### 5.2.4 查询1项以上诊断结果不健康的病患的编号及其平均诊断结果评分

SQL> select Pno, avg(Result) from Result

2 group by Pno having Pno in(

3 select Pno from Result

4 where Result<60

5 group by Pno

6 having count(\*) >1);

#### 5.2.5 查询类别信息的同时显示其所在科室名称

SQL> select c.Cateid,c.DepPname,m.DeparDname

2 from Cate c left join Depart m

3 on c.Departid=m.Departid;

#### 5.2.6 查询每种类别被患者选择的数目

SQL> select cno, count(\*) rs,

2 (select cname from project

3 where cno=Result.cno)

4 cname from Result

5 group by cno;

#### 5.2.7 查询所有病患的就诊信息

SQL> select s.Pno, s.Pname patdenDname,

2 c.cno, c.cname projectname

3 from patient s, Result g, project c

4 where s.Pno=g.Pno and g.cno=c.cno;

## 6 SQL语句

### 6.1 显示表中的编号为1的类别名

使用PL/SQL程序块，输出显示project表中的编号为1的类别名。

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 project where cno=id;

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

8 end;

9 /

1oracle

PL/SQL 过程已成功完成。

### 6.2 判断诊断结果61分所处的等级

在PL/SQL中，使用if条件语句判断诊断结果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('健康');

12 else

13 dbms\_output.put\_line('不健康');

14 end if;

15 end;

16 /

健康

### 6.3 判断患者的诊断结果是否健康

在PL/SQL中，查询所有患者的诊断结果是否有不健康，如有不健康就触发异常并输出。

SQL> declare

2 cursor c1 is select Pname from Result where Result<60;

3 one Result.Pname%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||'不健康');

13 close c1;

14 end;

15 /

杨不健康

PL/SQL 过程已成功完成。

## 7 创建存储过程

### 7.1 创建一个存储过程update\_patient

创建一个存储过程update\_patient，该过程用来将patient表中的编号为1的患者的姓名改为”惨惨”。

SQL> create procedure update\_patient

2 as

3 begin

4 update patient set Pname='惨惨' where Pno =1;

5 end update\_patient;

6 /

过程已创建。

使用execute语句调用存储过程，如下：

SQL> execute update\_patient;

PL/SQL 过程已成功完成。

### 7.2 创建一个存储过程get\_Result\_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\_Result\_information

2 (s\_no number)

3 as

4 s\_name varchar2(10);

5 c\_no number;

6 c\_name varchar2(20);

7 s\_Result number(3);

8 c\_credit number(3);

9 begin

10 select Pname,cno,cname,Result,credit

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

12 from Result where Pno=s\_no;

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

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

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

16 DBMS\_OUTPUT.PUT\_LINE('诊断结果：'||s\_Result);

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

18 end get\_Result\_information;

19 /

过程已创建。

调用get\_Result\_information存储过程。例如获取Pno为1401的患者的成绩信息，如下：

SQL> set serveroutput on

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

患者姓名：杨

类别编号：2

类别名称：java

诊断结果：40

指标：8

PL/SQL 过程已成功完成。

## 8 函数设计

### 8.1 创建一个函数get\_Pname

创建一个函数get\_Pname，该函数实现按Pno获取Pname，函数创建如下：

SQL> create function get\_Pname(pat\_num number)

2 return varchar2 as

3 pat\_name patient.Pname%type;

4 begin

5 select Pname into pat\_name from patient where Pno=pat\_num;

6 return pat\_name;

7 end get\_Pname;

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\_Pname(1) from dual;

GET\_PNAME(1)

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

shen

### 8.2 创建一个函数re\_pat\_info

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

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

2 v\_result varchar2(100);

3 cursor cur\_Cate is

4 select a.DepPname as DepPname,

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

6 from Cate a

7 inner join patient b

8 on a.Cateid = b.Cateid

9 inner join Result c

10 on c.Pno = b.Pno

11 where a.Cateid = cl\_id

12 group by a.DepPname;

13 c\_row cur\_Cate%rowtype;

14 begin

15 for c\_row in cur\_Cate loop

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

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 Cateid from Cate;

5 begin

6 open cur;

7 fetch cur into v\_number;

8 while cur%found loop

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

10 fetch cur into v\_number;

11 end loop;

12 close cur;

13 end;

14 /

## 9 备份方案

### 9.1 RMAN（备份与恢复管理器）

#### 9.1.1 为目录创建一个单独的表空间

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

#### 9.1.2创建RMAN用户

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

#### 9.1.3 给RMAN授予权限

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

#### 9.1.4 打开RMAN

$>RMAN

#### 9.1.5 连接数据库

RMAN>connect catalog rman/rman

#### 9.1.6 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 1856492365

1 row selected.

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

SQL> EXECUTE dbms\_rcvcat.unregisterdatabase(1 , 1856492365)

（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;