# Oracle

## 主流数据库

6种

小型:access

中型:sql server,mysql,informix

大型:oracle,Sybase,db2

数据库的选择:

项目规模:a,负载量，用户量

b,成本

c,安全性

小型:负载小，100人以内，千元以下，对安全性要求不高

中型:日访问量5000-15000以内，万元以下，如商务网站

大型:负载可以处理海量数据，sybase<oracle<db2

## 安装

全局数据库:自己指定一个数据库，数据库名称不超过8个字符

字符集:选择简体中文

自动生成两个用户，sys和system

a,sys是超级用户，具有sysdba角色，有createdatabase权限，默认密码是manager

b,system是管理操作员，具有sysoper角色，无createdatabase权限，默认密码是change\_on\_install

c,一般，对数据库进行维护，使用system用户登录就可以

oracle

一个实例对应一个数据库，一个数据库可以拥有多个用户，用户权限不同而看到实例中不同对象，对象就是表。

Oracle工具:

A,在运行窗口输入sqlplusw

B,在运行窗口输入sqlplus

C,在开始菜单中打开enterprise manager console,进入图形界面

图形界面工具中

方案->表->用户，可以看表结构，可以看表数据

D,pl/sql工具

## 命令行常用命令

1,Conn jw\_user/jw\_user@orcl 普通用户连接数据库

Conn sys/zfsoft123@orcl as sysdba 当用特权用户登录时必须带上as sysdba或as sysoper

2,show user 查看当前用户名

3,conn sys/manager 切换用户

4,disc 断开与当前数据库的连接

5,passw 修改用户密码

修改其他用户的密码时通过sys/system用户

6,exit 断开与数据库的连接并退出sqlplus

7,start d:/aa.sql 或 @ d:/aa.sql 运行sql脚本

8,edit d:/aa.sql 编辑sql脚本

9,spool d:/bb.sql

Select \* from emp;

Spool off

将查询到的内容输出到指定文件中去

10,pl/sql中查看用户可以操作的表是在table文件下查看

11,文件->命令窗口

Sql> select \* from emp where ename=’smith’

替换符’&’

Selelct \* from emp where ename=’&name’

12,show linesize

Set linesize 90 设置显示行的宽度，默认是80

13,show pagesize

Set pagesize 15 设置每页显示的行数目，默认是14

## 用户管理

Create user lvbo identified by lvbo007 创建用户，只有管理员用户可以创建

用户密码首位是字母

新创建的用户没有任何权限

权限有两种，一种是系统权限，是用户对数据库的相关权限；一种是对象权限，是用户对其他用户的数据对象的操作权限

Password lvbo 修改用户密码

Drop user lvbo 删除用户，若用户已有创建的表，则须加参数cascade

Drop user lvbo cascade

Grant connect to lvbo 分配connect 角色权限给lvbo用户

角色有两种，一是预定义角色，常见的有connect角色，dba角色，resource角色，其中resource角色可以让用户在任何表空间建表；一是自定义角色

Grant select on emp to lvbo 使用户lvbo可以访问用户scott的emp表

Select \* from scott.emp 在用户lvbo中访问用户scott的emp表

在oracle中，用户scott的表称为方案

对象权限有select,update,insert,delete,all,create index

Grant update on emp to lvbo 使用户lvbo可以修改用户scott的emp表

Revoke select,update on emp from lvbo 用户scott收回lvbo的select,update权限

二次授权，lvbo可以把scott给他的权限授权给别人

## Profile用户口令管理

A,账户锁定

sql>Create profile lock\_account limit

failed\_login\_attenpts 3 password\_lock\_time 2;

sql>alter user scott profile lock\_account;

设置用户scott最多只能尝试登录3次，锁定时间为2天

B,给用户scott解锁

sql>alter user scott account unlock;

C,终止口令(作用是让用户定期更改密码)

Sql>creaete profile myprofile limit

password\_life\_time 10 password\_grace\_time 2;

sql>alter user scott profile myprofile;

设置用户scott，要求用户每隔10天修改自己的密码，期限2天

## 表的管理

表、列命名:

a,以字母开头， b,长度30字符以内，

c,不能使用oracle保留字， d,只能使用A-Z,a-z,0-9,$,#

数据类型:

A,字符型

Char 定长，最大2000字符，效率高

Varchar2(20) 变长，最大4000字符，节省空间

Clob(character large object) 字符型大对象，最大4G

B,数字型

Number 小数，整数 ，-10^38—10^38

Number(5,2) 小数，-999.99—999.99

Number(5) 整数，-99999—99999

C,日期类型

Date 年月日时分秒

Timestamp 时间戳，对date的扩展

D,图片类型

Blob 二进制数据，可存储图片/声音,4G

表空间:

建表

Create table student (xh number(4),

Xm varchar(20),

Sex char(2),

Birthday date,

Sal number(7,2)

);

Create table classes(classid number(2),

Cname varchar2(40)

);

添加字段 alter table student add (classid number(2)

);

查询表结构 desc student

修改字段长度 alter table student modify (xm varchar2(30)

);

修改字段类型或名字（不能有数据）alter table student modify(xm char(30)

);

删除一个字段 alter talble student drop column sal;

修改表的名字 rename student to stu;

删除表 drop table student

## 数据操作

添加数据 insert into student values(1,’小明’,’男’,’11-12月-1997’,12);

修改日期的默认格式 alter session set nls\_date\_format=’yyyy-mm-dd’;

插入部分字段 insert into student(xh,xm,sex) values(‘a001’,’john’,’女’);

插入空值 insert into student (xh,xm,sex,birthday) values (‘a002’,’matin’,’男’,null);

查询记录中某个字段为空的记录

Select \* from student where birthday is (not) null;

修改单个字段 update student set sex=’女’ where xh=’a001’;

修改多个字段 update student set sex=’男’,sal=sal/2,birthday=’1980-04-02’ where xh=’a002’;

删除数据

A,delete from student 删除表中记录，表结构还在，还可以写日志，可恢复

oracle回滚

sql>save point aa; 设置保存点aa

sql>delete from student;

sql>rollback to aa; 回滚到保存点aa, 可以有多个保存点

B,drop table student 删除表结构和数据

C,truncate table student 删除表中记录，表结构还在，不可以写日志，不可恢复，删除速度快

## 查询

清屏 clear;

查看表结构 desc student;

打开操作时间开关set timing on

复制表记录 insert into users(userid,username,userpass) select \* from users;

查询表中总记录数 select count(\*) from users;

取消查询结果中的重复记录 select distinct deptno,job from emp;

重复记录指记录内容完全相同

Oracle中的记录内容区分大小写

1，查询每个雇员的年工资(只有月工资)

Select ename ‘姓名’,sal\*12 as ‘年薪’ from emp;

2, 查询每个雇员的年工资(月工资+奖金)

Select ename ‘姓名’,sal\*12+nvl(comm,0)\*13 as ‘年薪’ from emp;

nvl(comm,0)\*13 as ‘年薪’ 当无奖金时按0计算，当有奖金时按奖金计算

3，查询1982年1月1日之后入职的员工（注意日期格式是固定的）

Select ename,hiredate from emp where hiredate>’1-1月-1982’;

4,like查询

Select ename,hiredate from emp where ename like ‘%O\_’;

5,in查询

Select \* from emp where empno in(123,124,125);

6，查询无上级的雇员

Select \* from emp where mgr is null;

7,查询工资大于500或者职位为manager,并且雇员首字母为J的雇员

Select \* from emp where (sal>500 or job=’manager’) and ename like ‘J%’;

8,表的复制

create table tempkcxx as select \* from jw\_jh\_kcdmb

9,按从高到低的顺序显示查询结果

Select \* from emp order by sal desc|asc

10,按部门号升序并且部门中员工薪水降序显示查询结果

Select \* from emp order by empno asc,sal desc;

11，按列的别名年薪降序排列查询结果(中文须加“”号)

Select ename,sal+(nvl(comm,0))\*12 “年薪” from emp order by “年薪” desc;

## 复杂查询

数据分组—函数max,min,avg,sum,count,

1,显示工资最高的员工信息

select ename,sal from emp where sal=(select max(sal) from emp)

2，查找工资高于平均工资的员工信息

Select \* from emp where sal>(select avg(sal) from emp)

3, 显示每个部门员工的平均工资，最大工资

Select empno,avg(sal),max(sal) from emp group by empno

4，显示每个部门每种岗位的最高工资、最低工资和平均工资

Select empno,job,min(sal),max(sal),avg(sal) from emp group by empno,job;

5,平均工资低于2000的部门号和平均工资

select avg(sal),empno from emp group by empno having avg(sal)>2000;

注:按部门分组，计算的是每个部门的平均工资，having是对分组后计算好的每个部门的信息再进行筛选。

6，顺序 group by,having,order by

7, 平均工资低于2000的部门号和平均工资并且按照从低到高的顺序显示

Select empno,avg(sal) from emp group by empno having avg(sal)>2000 order by avg(sal);

## 多表查询

多表查询中有一个笛卡尔集的概念，多表查询是要消除笛卡尔集现象

1，查询雇员的名字，薪水，部门名称

select a.ename,a.sal,b.dname from emp a,dept b where a.deptno=b.deptno;

2，查询雇员的名字，薪水和薪水级别

select a.ename,a,sal,b,grade from emp a,salgrade b where a.sal between b.losal and b.misal

3，自连接查询指对同一张表的查询.显示雇员和雇员的上级

Select worker.ename,boss.ename from emp worker,emp boss where worker.mgr=boss.empno;

4，正方jwglxt实操

select

a.xh\_id 学号,

a.xm 姓名,

c.xjztmc 学籍状态名称,

d.xjztmc 异动前学籍状态,

e.xjztmc 异动后学籍状态,

b.xwsj 行文时间

from

jw\_xjgl\_xsjbxxb a,

jw\_xjgl\_xjydb b,

JW\_XJGL\_XJZTDMB c,

JW\_XJGL\_XJZTDMB d,

JW\_XJGL\_XJZTDMB e

Where

a.xh\_id=b.xh\_id

and a.xjztdm=c.xjztdm

and b.ydqxjzt=d.xjztdm

and b.ydhxjzt=e.xjztdm

and a.xjztdm<>b.ydhxjzt

and b.xwsj<'2017-10-08'

## 子查询

1，单行子查询，指返回结果只有一行数据的子查询

显示和雇员smith在同一个部门的所有雇员

Select \* from emp where empno=(select deptno from emp where ename=’smith’);

数据库执行语句是从右往左，注意这一点可以提升sql速度

2，多行子查询，指返回结果有多行数据的自查询

显示10号部门有的职位的所有雇员

Select \* from emp where job in (select distinct job from emp where deptno=10)

3,显示薪水比30号部门所有雇员的薪水都高的雇员的姓名，薪水和部门号

1,Select ename,sal,deptno from emp where sal>all (select sal from emp where deptno=30)

2,select \* from emp where sal>(select max(sal) from emp where deptno=30)

4, 显示薪水比30号部门任意一个雇员的薪水都高的雇员的姓名，薪水和部门号

1,Select ename,sal,deptno from emp where sal>any (select sal from emp where deptno=30)

2, select \* from emp where sal>(select min(sal) from emp where deptno=30)

## 多列子查询

1,显示和smith部门号和岗位全部相同的雇员

Select \* from emp where (deptno,job)=( Select deptno,job from emp where ename=’smith’);

2,显示薪水高于自己部门平均薪水的雇员

1,查自己部门的平均薪水

select deptno,avg(sal) mysal from emp group by deptno;

2，把上面的结果看成一张表

Select \* from emp a,( Select deptno,job from emp where ename=’smith’) b where a.deptno=b.deptno and a.sal>b.mysal

当在from使用子查询时，必须指定别名此时不要加as

## 分页查询

一，Rownum分页（内嵌视图）

1，

(Select \* from emp)

2,显示rownum[oracle分配的]

Select a1.\*,rownum rn from (select \* from emp) a1

3,二分查询，速度比2000，mysql快

Select \* from (Select a1.\*,rownum rn from (select \* from emp) a1 where rownum<=10) where rn>=6;

4,几个查询变化

A,指定列，只需改最里层的\*

Select \* from (Select a1.\*,rownum rn from (select ename,sal from emp) a1 where rownum<=10) where rn>=6;

B,排序，只需改最里层的\*

Select \* from (Select a1.\*,rownum rn from (select \* from emp order by sal asc) a1 where rownum<=10) where rn>=6;

二，根据rowid分页(比方法一快)

Select \* from t\_xiaoxi where rowed in (select rid from (select rownum rn,rid from select rowid,rid,cid from t\_xiaoxi order by cid desc) where rownum<10000) where rn>9980) order by cid>desc

1,查询表有多少条记录

Select count(\*) from emp;

## 其他子查询

1,用查询结果创建新表

Create table myemp (id,name,sal) as select empno,ename,sal from emp;

2,合并查询（查询速度比用and,or快）

Union(获得两个结果集的并集)

Select ename,sal,job from emp where sal>3000

Union

Select ename,sal,job from emp where job=’manager’

Union all(都罗列出来，不取消重复行)

Intersact(获得两个结果的交集)

Minus(前者的结果集减去后者的结果集之后的结果集)

## 创建数据库的方法

1,通过oracle提供的向导工具dbca[数据库配置助手]

2,通过手工步骤直接创建

## Pl/sql

Pl/sql的作用

提高应用程序的运行性能

模块化的设计思想

减少网络传输量

提高安全性

Pl/sql的缺点

移植性不好

Sqlplus开发工具 （oracle公司提供的工具）

简单案例

编写存储过程，过程可以向某表中添加记录

1,创建表

Create table mytest(name varchar(20),passwd varchar2(30))

2,创建过程

Create or replace procedure lvbo\_pro1 is

Begin

Insert into mytest values(‘lvbo’,’lvbo’);

End;

/

如何查看错误信息

Show error

如何调用该过程

1,exec 过程名（参数值1，2…）

2,call 过程名（参数值1，2…）

Developer开发工具（集成开发环境）

简单案例

编写存储过程，过程可以删除某表中记录

Create or replace procedure lvbo\_pro2 is

Begin

Delete from mytest where name=’lvbo’;

End;

/

Pl/sql基础

单位:块（编程）==编写过程、函数、触发器、包

编写规范:

1,注释

单行注释 –

Select \* from emp where empno=7708;--取得员工信息

多行注释

/\* …\*/

2,标示符号的命名规范

定义变量时，建议用v\_作为前缀 v\_sal

定义常量时，建议用c\_作为前缀 c\_rate

定义游标时，建议用 \_cursor作为后缀， emp\_cursor

定义例外时，建议用e\_作为前缀，e\_error

块

Pl/sql的基本程序单元，编写pl/sql就是编写块。

简单功能需要一个块，复杂功能可能需要一个块中嵌套其他的块。

块由定义部分declare,执行部分begin,例外处理部分Exception组成

如:

Declare

Begin

Exception

End;

## 块

1,简单的块，只有执行部分

set serveroutput on/off --打开输出选项

begin

dbms\_output.put\_line(‘hello world’);

end;

/

Dbms\_output是oracle提供的包，包里含有过程和函数

2，只有定义部分，执行部分的块

Declare

V\_ename varchar2(5);

V\_sal number(7,2);

Begin

Select ename,sal into v\_ename,v\_sal from emp where empno=&aa;

Dbms\_output.put\_line(‘用户名是’||v\_ename||’薪水’||v\_sal);

End;

/

3，包含定义部分，执行部分和例外处理部分的块

Declare

V\_ename varchar2(5);

V\_sal number(7,2);

Begin

Select ename,sal into v\_ename,v\_sal from emp where empno=&aa;

Dbms\_output.put\_line(‘用户名是’||v\_ename||’薪水’||v\_sal);

Exception

When no\_data\_found then

Dbms\_output.put\_line(‘编号输入错误，请重新输入’);

End;

/

## 过程

1,

Create procedure sp\_pro1(spName varchar2,newSal number) is

Begin

--根据用户名修改工资

Update emp set sal=newSal where ename=spName;

End;

/

调用:

窗口:

exec;call

java中:

## 函数

1，

--输入雇员名，返回雇员年薪

Create function sp\_fun1(spName varchar2)

return number is yearSal number(7,2);

begin

select sal\*12+nvl(comm.,0)\*12 into yearSal from emp

where ename=spName;

return yearSal;

end;

/

调用:

Sql>Var aaa number;

Sql>call sp\_fun1(‘scott’) into:aaa;

## 包

1，包由包规范和包体两部分组成

--创建包

Create package sp\_package is

Procedure update\_sal(name varchar2,newSal number);

Function annual\_income(name varchar2) return number;

End;

/

--创建包体

Create or replace package body sp\_package is

Procedure update\_sal(name varchar2,newsal number) is

Begin

Update emp set sal=newsal where ename=name;

End;

Function annual\_income(name varhcar2)

return number is annual\_salary number;

begin

select sal\*12+nvl(comm,0) into annual\_salary from emp where ename=name;

return annual\_salary;

end;

End;

/

调用:

Exec sp\_package.update\_sal(‘scott’,120);

## 触发器

当定义触发器时，必须要指定触发的事件和触发的操作，常用的触发事件包括insert,update,delete语句，而触发操作实际就是一个pl/sql块。可以使用create trigger来建立触发器。

## 变量

类型:

标量类型（scalar）

复合类型（composite）

参照类型(reference)

Lob(large object)

标量

1,定义一个变长字符串

V\_ename varchar2(10);

2,定义一个小数，-9999.99~9999.99

V\_sal number(6,2);

3,定义一个小数并给一个初始值5.4

V\_sal2 number(6,2) :=5.4;

4,定义一个日期类型

V\_hiredate date;

5,定义一个布尔变量，不能为空，初始值为false

V\_valid boolean not null default false;

案例:

1,

--输入雇员号，显示雇员名，工资，个人所得税（税率为0.03）

Declare

C\_tax\_rate number(3,2) :=0.03;

V\_ename varchar2(5);

V\_sal number(7,2);

V\_tax\_sal number(7,2);

Begin

Select ename,sal into v\_ename,v\_sal from emp where empno=&no;

V\_tax\_sax :=v\_sal\*c\_tax\_rate;

Dbms\_output.put\_line(‘姓名是’||v\_ename||’工资:’||v\_sal||’交税:’||v\_tax\_sal);

End;

/

2,

--输入雇员号，显示雇员名，工资，个人所得税（税率为0.03）

Declare

C\_tax\_rate number(3,2) :=0.03;

V\_ename emp.ename%type; --一种标量类型，大小随ename定

V\_sal number(7,2);

V\_tax\_sal number(7,2);

Begin

Select ename,sal into v\_ename,v\_sal from emp where empno=&no;

V\_tax\_sax :=v\_sal\*c\_tax\_rate;

Dbms\_output.put\_line(‘姓名是’||v\_ename||’工资:’||v\_sal||’交税:’||v\_tax\_sal);

End;

/

复合变量

用于存放多个值的变量，主要包括:1,2用的多

1,pl/sql记录 类似于结构体或类

2，pl/sql表 类似数组，但下标可以为负数

3,嵌套表

4,varray(动态数组)

实例

1,pl/sql记录

Declare

--定义一个pl/sql记录类型 emp\_record\_type,

--类型包含三个数据,name,salary,title

Type emp\_record\_type is

record(name emp.ename%type,salary emp.sal%type,title emp.job%type);

--定义一个sp\_record变量，变量类型是emp\_record\_type

Sp\_record emp\_record\_type;

Begin

Select ename,sal,job into sp\_record from emp where empno=7788;

Dbms\_output.put\_line(‘员工名’||sp\_record.name||’工资:’||sp\_record.salary);

End;

/

2，pl/sql表

Declare

--定义了一个pl/sql表类型sp\_table\_type,

--该类型用于存放emp.ename%type

-- index by binany\_integer表示下标为整数，按整数排序，可为负

Type sp\_table\_type is table of emp.ename%type index by binany\_integer;

--定义一个sp\_table变量，变量类型是sp\_table\_type

Sp\_table sp\_table\_type;

Begin

Select ename into sp\_table(0) from emp where empno=7788;

Dbms\_output.put\_line(‘员工名:’||sp\_table(0));

End;

/

参照变量

用于存放数值指针的变量。

两种:游标变量（ref cursor）和对象类型变量(ref obj\_type)。

游标变量

定义游标时不需要指定相应的select语句，

但当使用游标时（open时）需要指定select语句，

这样一个游标就与一个select语句结合了。

实例:

1,编写一个块，输入部门号，显示该部门所有员工的姓名和工资

Declare

--定义游标类型

Type sp\_emp\_cursor is ref cursor;

--定义一个游标变量

Test\_cursor sp\_emp\_cursor;

--定义变量

V\_ename emp.ename%type;

V\_sal emp.sal%type;

Begin

--执行

--把test\_cursor和一个select结合

Open test\_cursor for select ename,sal from emp where deptno=&no;

--循环取出

Loop

Fetch test\_cursor into v\_ename,v\_sal;

--判断工资高低，确定是否更新

--判断test\_cursor是否为空

Exit when test\_cursor%notfound;

Dbms\_output.put\_line(‘名字:’||v\_ename||’工资’||v\_sal);

End loop;

End;

/

## Pl/sql进阶

控制结构

If语句、循环语句、控制语句goto和null

If案例

1,编写过程，输入雇员名，若工资低于2000，就增加10%工资

Create or replace procedure sp\_pro6(spName varchar2) is

--定义

V\_sal emp.sal%type;

Begin

--执行

Select sal into v\_sal from emp where ename=spName;

--判断

If v\_sal<2000 then

Update emp set sal=sal\*1.1 where ename=spName;

End if;

End;

/

2, 编写过程，输入雇员名，若补助不是0就增加100，若为0就设为200

Create or replace procedure sp\_pro6(spName varchar2) is

--定义

V\_comm emp.comm%type;

Begin

--执行

Select comm into V\_comm from emp where ename=spName;

--判断

If V\_comm <>0 then

Update emp set comm=comm+100 where ename=spName;

Else

Update emp set comm=comm+200 where ename=spName;

End if;

End;

/

3,输入雇员号，若为president,工资增加2000；若为manager,工资增加500；其他增加200

Create or replace procedure sp\_pro(spNo number) is

--定义

V\_job emp.job%type;

Begin

--执行

Select job into v\_job from emp where empno=spNo;

If v\_job=’president’ then

Update emp set sal=sal+1000 where empno=spNo;

Elsif v\_job=’manager’ then

Update emp set sal=sal+500 where empno=spNo;

Else

Update emp set sal=sal+200 where empno=spNo;

End if;

End;

/

循环案例

1,编写过程，输入用户名，循环添加10个用户到表中，用户编号从1开始增加

Create or replace procedure sp\_spro(spName varchar2) is

--定义

V\_num number :=1;

Begin

Loop

Insert into users values(v\_num,spName);

--判断是否退出循环

Exit when v\_num=10;

--自增

V\_num :=v\_num+1;

End loop;

End;

/

2,编写过程，输入用户名，循环添加10个用户到表中，用户编号从11开始增加

Create or replace procedure sp\_spro(spName varchar2) is

--定义

V\_num number :=11;

Begin

While v\_num<=20 loop

--执行

Insert into users values(v\_num,spName);

V\_num :=v\_num+1;

End loop;

End;

/

3,for循环基本结构

Begin

For i in reverse 1 .. 10 loop

Insert into users values(I,’顺平’);

End loop;

End;

/

顺序控制语句goto和null案例

1，Goto语句用于跳转到特定标号去执行语句

Declare

I int :=1;

Begin

Loop

Dbms\_output.put\_line(‘输出i=’||i);

If i=10 then

Goto end\_loop;

End if;

I :=i+1;

End loop;

<<end\_loop>>

Dbms\_output.put\_line(‘循环结束’);

End;

/

2，null用于提高可读性

Declare

V\_sal emp.sal%type;

V\_ename emp.ename%type;

Begin

Select ename,sal into v\_ename,v\_sal from emp where empno=&no;

If v.sal<3000 then

Update emp set comm=sal\*0.1 where ename=v\_ename;

Else

Null;

End if;

End;

/

## 分页过程

案例

1,无返回值的过程

--创建表 book

Create table book (bookid number,bookname varchar2(50),publishhouse varhcar2(50));

--编写过程

--in:表示一个输入参数

--out:表示一个输出参数

Create or replace procedure sp\_pro7(spBookid in number,

SpbookName in varchar2,

sppublishHouse in varchar2) is

begin

insert into book values(spBookId,spbookName,sppublishHouse);

end;

/

--在java中调用一个无返回值的过程

//引入

Import java.sql.\*;

Try{

//1、加载驱动

Class.forName(“oracle.jdbc.driver.OracleDriver”);

Connection ct=DriverManager.getConnection(

“jdbc:oracle:thin:@127.0.0.1:1521:MYORA1”,

”scott”,”m123”);

//2、创建CallableStatement

CallableStatement cs=ct.prepareCall(“{call sp\_pro7(?,?,?)}”);

//3、给？赋值

Cs.setInt(1,10);

Cs.setString(2,”笑傲江湖”);

Cs.setString(3,”人民出版社”);

//4、执行

Cs.execute();

//5、关闭资源

}catch (Exception e){

e.printStackTrace();

}finally{

//5、关闭各个打开的资源

}

2,有返回值的过程(单个记录)

--编写过程，输入雇员编号，返回该雇员姓名

--有输入和输出的过程

Create or replace procedure sp\_pro8(

Spno in number,

spName out varchar2) is

begin

select ename into spName from emp where empno=spno;

end;

/

--在java中调用一个有返回值的过程

//引入

Import java.sql.\*;

Try{

//1、加载驱动

Class.forName(“oracle.jdbc.driver.OracleDriver”);

Connection ct=DriverManager.getConnection(

“jdbc:oracle:thin:@127.0.0.1:1521:MYORA1”,

”scott”,”m123”);

//2、创建CallableStatement

CallableStatement cs=ct.prepareCall(“{call sp\_pro8(?,?)}”);

//3、给？赋值

Cs.setInt(1,7788);

Cs.registerOutParameter(2,oracle.jdbc.OracleTypes.varchar);

//4、执行

Cs.execute();

//5、取出返回值

//注意？顺序

String name=cs.getString(2);

System.out.println(“编号是7788的名字”+name);

//5、关闭资源

}catch (Exception e){

e.printStackTrace();

}finally{

//5、关闭各个打开的资源

}

--输入雇员号，显示雇员姓名、工资、工作

Create or replace procedure sp\_pro8(

Spno in number,

spName out varchar2,

spSal out number,

spJob out varchar2) is

begin

select ename,sal,job into spName,spSal,spJob from emp where empno=spno;

end;

/

--在java中调用一个有返回值的过程

//引入

Import java.sql.\*;

Try{

//1、加载驱动

Class.forName(“oracle.jdbc.driver.OracleDriver”);

Connection ct=DriverManager.getConnection(

“jdbc:oracle:thin:@127.0.0.1:1521:MYORA1”,

”scott”,”m123”);

//2、创建CallableStatement

CallableStatement cs=ct.prepareCall(“{call sp\_pro8(?,?,?,？)}”);

//3、给？赋值

Cs.setInt(1,7788);

Cs.registerOutParameter(2,oracle.jdbc.OracleTypes.varchar);

Cs.registerOutParameter(3,oracle.jdbc.OracleTypes.double); Cs.registerOutParameter(4,oracle.jdbc.OracleTypes.varchar);

//4、执行

Cs.execute();

//5、取出返回值

//注意？顺序

String name=cs.getString(2);

String job=cs.getString(4);

System.out.println(“编号是7788的名字:”+name+

“工作岗位:”+job);

//5、关闭资源

}catch (Exception e){

e.printStackTrace();

}finally{

//5、关闭各个打开的资源

}

3,有返回值的过程(多个记录/结果集)，必须用package

--1、创建一个包package

Create or replace package testpackage as

Type test\_cursor is ref cursor;

End testpackage;

/

--2、创建过程

Create or replace procedure sp\_pro9(spNo in number,

P\_cursor out testpacdage.test\_cursor) is

Begin

Open p\_cursor for select \* from emp where deptno=spNo;

End;

/

--在java中调用一个有返回值的过程

//引入

Import java.sql.\*;

Try{

//1、加载驱动

Class.forName(“oracle.jdbc.driver.OracleDriver”);

Connection ct=DriverManager.getConnection(

“jdbc:oracle:thin:@127.0.0.1:1521:MYORA1”,

”scott”,”m123”);

//2、创建CallableStatement

CallableStatement cs=ct.prepareCall(“{call sp\_pro8(?,?)}”);

//3、给？赋值

Cs.setInt(1,10);

Cs.registerOutParameter(2,oracle.jdbc.OracleTypes.varchar);

Cs.registerOutParameter(2,oracle.jdbc.OracleTypes.cursor);

//4、执行

Cs.execute();

//5、取出返回值

//注意？顺序

ResultSet rs=(ResultSet)cs.getObject(2);

While(rs.next()){

System.out.println(rs.getInt(1)+””+rs.getString(2));

}

//5、关闭资源

}catch (Exception e){

e.printStackTrace();

}finally{

//5、关闭各个打开的资源

}

--oracle的分页

--1

Select t1.\*,rownum rn from (select \* from emp) t1;

--2

Select t1.\*,rownum rn from (select \* from emp) t1 where rownum<=10;

--3

Select \* from

(Select t1.\*,rownum rn from (select \* from emp) t1 where rownum<=10)

Where rn>=6;

--4 开发一个包，定义类型test\_cursor,是个游标

Create or replace package tespackage as

Type test\_cursor is ref cursor;

End tespackage;

--5 开始编写分页的过程

Create or replace procedure fenye

(tableName in varchar2,

Pagesize in number, --每页记录数

pageNow in number, --当前页

myrows out number, --总记录数

myPageCount out number, --总页数

p\_cursor out testpackage.test\_cursor –返回的记录集

) is

--定义部分

--定义sql语句 字符串

V\_sql varchar2(1000);

--定义两个整数

V\_begin number :=(pageNow-1)\*Pagesize+1;

V\_end number pageNow\*Pagesize;

Begin

--执行部分

V\_sql :=’ Select \* from

(Select t1.\*,rownum rn from (select \* from ‘|| tableName ||’ order by sal) t1 where rownum<=’||v\_end||’)

Where rn>=’||v\_begin||’;

’;

--把游标和sql语句关联

Open p\_cursor for v\_sql;

--计算myrows和myPageCount

--组织一个sql

V\_sql :=’select count(\*) from ‘||tableName;

--执行sql,并把返回的值，赋值给myrows

Execute immediate v\_sql into myrows;

--计算myPageCount

If mod（myrows,Pagesize）=0 then

myPageCount :=myrows/Pagesize;

else

myPageCount :=myrows/Pagesize+1;

end if;

--关闭游标

Close p\_cursor;

End;

/

--在java中调用一个有返回值的过程

//引入

Import java.sql.\*;

Try{

//1、加载驱动

Class.forName(“oracle.jdbc.driver.OracleDriver”);

Connection ct=DriverManager.getConnection(

“jdbc:oracle:thin:@127.0.0.1:1521:MYORA1”,

”scott”,”m123”);

//2、创建CallableStatement

CallableStatement cs=ct.prepareCall(“{call fenye(?,?,?,?,?,?,?)}”);

//3、给？赋值

Cs.setString(1,”emp”);

Cs.setInt(2,5);

Cs.setInt(3,1);

Cs.registerOutParameter(4,oracle.jdbc.OracleType.interger);

Cs.registerOutParameter(5,oracle.jdbc.OracleType.interger);

Cs.registerOutParameter(6,oracle.jdbc.OracleType.cursor);

//4、执行

Cs.execute();

//5、取出返回值

Int rowNumber=cs.getInt(4);

Int pageCount=cs.getInt(5);

resultSet rs=(ResultSet)cs.getObject(6);

//显示

System.out.println(“rowNum”+rowNum);

System.out.println(“总页数:”+pageCount);

While(rs.next()){

System.out.println(“编号:”+rs.getInt(1)+”名字:”+rs.getString(2)+”薪水:”+rs.getFloat(6));

}

//5、关闭资源

}catch (Exception e){

e.printStackTrace();

}finally{

//5、关闭各个打开的资源

}

## 例外处理

三种:预定义例外、非预定义例外、自定义例外

案例

1,编写过程，接收雇员编号，并显示雇员姓名。若雇员编号不存在，怎么处理

Declare

V\_ename emp.ename%type;

Begin

Select \* from emp where empno=&gno;

Dbms\_output.put\_line(‘名字:’||v\_ename);

Exception

When no\_data\_found then --预定义例外

Dbms\_output.put\_line(‘编号没有’);

End;

/

2,case\_no\_found --预定义例外

在编写case语句时，若when子句中没有包含必须的条件分支，就会触发case\_not\_found例外

Create or replace procedure sp\_pro6(spno number) is

V\_sal emp.sal%type;

Begin

Select sal into v\_sal from emp where empno=spno;

Case

When v\_sal<1000 then

Update emp set sal=sal+100 where empno=spno;

When v\_sal<2000 then

Update emp set sal=sal+200 where empno=spno;

End case;

Exception

When case\_not\_found then

Dbms\_output.put\_line(‘case语句没有与’||’相匹配的条件’);

End;

/

3,cursor\_already\_open --预定义例外

当重新打开已经打开的游标时，触发cursor\_already\_open

Declare

Cursor emp\_cursor is select ename,sal from emp;

Begin

Open emp\_cursor;

For emp\_record1 in emp\_cursor loop –in又打开一次游标

Dbms\_output.put\_line(emp\_record1.ename);

End loop;

Exception

When cursor\_already\_open then

Dbms\_output.put\_line(‘游标已经打开’);

End;

/

4,dup\_val\_on\_index --预定义例外

在唯一索引对应的列上插入重复的值时，触发例外

Begin

Insert into dept values(10,’公关部’,’北京’);

Exception

When dup\_val\_on\_index then

Dbms\_output.put\_line(‘在deptno列上不能出现重复值);

End;

5,invalid\_cursor –预定义例外

当试图在不合法的游标上执行操作时，触发例外

例如:从未打开的游标取数据，或关闭未打开的游标

Declare

Cursor emp\_cursor is select ename,sal from emp;

Emp\_record emp\_cursor%rowtype;

Begin

--open emp\_cursor;--打开游标

Fetch emp\_cursor into emp\_record;

Dbms\_output.put\_line(emp\_record.ename);

Close emp\_cursor;

Exception

When invalid\_cursor then

Dbms\_output.put\_line(‘请检查游标是否打开’);

End;

/

6,invalid\_number –预定义例外

当输入的数据有误时，触发例外

Begin

Update emp set sal=sal+’1oo’;

Exception

When invalid\_number then

Dbms\_output.put\_line(‘输入的数字不正确’);

End;

/

7,too\_many\_rows –预定义例外

当执行select into 语句时，若返回超过了一行，触发例外

Declare

V\_ename emp.ename%type;

Begin

Select ename into v\_ename from emp;

Exception

When too\_many\_rows then

Dbms\_output.put\_line(‘返回了多行’);

End;

/

8、zero\_divide –预定义例外

当执行2/0 语句时，若返回超过了一行，触发例外

9、value\_error –预定义例外

当在执行赋值操作时，若变量长度不足以容纳实际数据，则触发例外

Declare

V\_ename varchar2(5);

Begin

Select ename into v\_ename from emp where empno=&nol;

Dbms\_output.put\_line(v\_ename);

Exception

When value\_error then

Dbms\_output.put\_line(‘变量长度不够’);

End;

/

10、其他预定义例外

--login\_denide

当用户非法登录时，触发例外

--not\_logged\_on

若用户没有登录就执行dml操作，触发例外

--storage\_error

若超出了内存空间或内存被损坏，触发例外

--timeout\_on\_resource

若oracle在等待资源时，出现超时则触发例外

自定义例外

案例

1,编写块，接收一个雇员编号，并给雇员工资增加1000元，若雇员不存在，则提示

Create or replace procedure ex\_test(spNo number) is

--定义一个例外

Myex exception;

Begin

--更新用户sal

Update emp set sal=sal+1000 where empno=spNo;

--sql%notfound表示没有更新成

--raise myex 表示触发例外

If sql%notfound then

Raise myex;

End if;

Exception

When myex then

Dbms\_output.put\_line(‘没有更新任何用户’);

End;

/

## 视图

视图是一个虚拟表，其内容由查询定义，同真实的表一样，视图包含一系列带有名称的列和行数据，但是，视图并不在数据库中以存储的数据值集形式存在。行和列数据来自由定义视图的查询所引用的表，并且在引用视图时动态生成。

视图与表的区别

--表需要占用磁盘空间，视图不需要

--视图不能添加索引

--使用视图可以简化复杂查询

--视图利于提高安全性，比如不同用户查看不同视图

案例

创建视图（视图之间可联合查询）

1,把emp表的sal<1000的雇员映射到视图

Create view myview as select \* from emp where sal<1000

2,为简化操作，用视图解决雇员编号，姓名和部门名称

Create view myview2 as select emp.empno,emp.ename,dept.dname from emp,dept where emp.deptno=dept.deptno;

创建或修改视图

Create or replace view myview3 as select \* from emp with read only

删除视图

Drop view myview3