第一部分(基本概况)

1. JDBC是什么?

Java官方推出一种用于访问数据库的通用技术(一堆接口).由各个数据库厂商提供实现.

2. JDBC标准

Driver　　加载JDBC实现

java.sql.Connection　建立连接

java.sql.Statement　封装SQL

java.sql.ResultSet　封装结果

3. (JDBC是JAVA平台访问关系型数据库的API)

4．常用的访数据库的编程接口主要有ODBC和JDBC,其中ODBC是基于C语言API,JDBC是基于JAVA的API,JDBC和ODBC的结构相仿.

5 .JDBC编程接口主要分布在JDK的两个包中

1、　　java.sql　-JDBC基本特征包

2、　　javax.sql – JDBC扩展功能包

6.JDBC应用程序的基本步骤.

1,注册驱动

Class.forName(“com.mysql.jdbc.Driver”)

2,建立数据库联接　Connection conn=DriverManager.*getConnection*(*url*,*user*,*password*);

3,创建数据库操作对象昂　PreparedStatement stmt=conn.createStatement()

4,执行SQL　ResultSet　rs=

conn.executeQuery(sql);

5处理结果集　while(rs.next()){}

6,关闭JDBC对象

(由于ResultSet对象和Statement对象都是由连接的对象,所以这三种类型的对象的关闭顺序应该是ResultSet　Statement　Connection)

一　url

="jdbc:oracle:thin:@192.168.0.26:1521:tarena","openlab","open123");

delect　update　insert　返回的整数(几条记录操作成功)

select返回的是结果集

7、将中文数据插入到数据库

step1 要保证数据库正确地设置了字符集。

step2 要保证 jdbc 驱动程序正确地进行了编码和解码。

jdbc:mysql://localhost:3306/jd1206db?useUnicode=true&characterEncoding=utf8

告诉 jdbc 驱动程序当前数据库的字符集。因为某些版本的驱动程序不能够正确地获知数据库的字符集设置。

8、mysql数据库的使用

(1)登录数据库

mysql -uroot;

(2)查看当前所有的数据库实例

show databases;

(3)创建一个新的数据库实例,并且设置了默认的字符集。

Create database huan

default character set utf8;

(4)使用某个数据库实例

use jd1207db;

(5) 查看当前数据库实例当中有哪些表

show tables;

(5)建表

```
create table t_emp(
    id int primary key auto_increment,
    name varchar(32),
    salary double,
    birthday date,
    age int
);
```

auto_increment:自增长列，当插入记录时，数据库自动生成一个唯一的值作为主键。

(6)插入记录

insert into t_emp(name,salary,age, birthday) values('zs',2000,22, '1999-2-2');

(7)查询、修改、删除记录

select * from t_emp;

update t_emp set age=23 where name='zs';

update t_emp set name=?, salary=?, age=? Where id=?

delete from t_emp where id=1;

(8)显示数据表

desc[table name]

(9)删除数据表

drop table[table name]

10)退出 mysql

Exit

(11)连接数据库

mysql -h localhost -uroot -p

(mysql -uroot -pmysqladmin)

Oracle（day1）

1 概念问题

SQL structured query language(结构化查询语言)

文件 : FILE

DATABASE（数据库）

DDL(data definition　language 数据定义语言　结构画表头)

create table（创建表）（column 列　数据类型　data type ，宽度 width　，constraint 约束）

alter　table（修改表结构）

drop table（删除数据）

DML　（data manipulation language 数据操作语言　数据 添行）

insert：插入数据

update：修改数据

delete：删除数据

TCL（transaction　control language 事物控制语言）

commit（提交）

rollback（回滚）

DQL　（data query language 数据查询语言） %%%%

Select:

DCL(data control language 数据控制语言)

grant（授权）

revoke（回收权限）

database object（数据库对象）

table（表）index（索引）view（视图）sequence（顺序号　序列号

++++++++

RDBMS（relationship database management system 数据库管理系统）

ORACLE ：　oracle 10g software my sql

IBM ：db2

MS ：SQL Server

－－－－－DBA(database administrator)

install oracle 10g（安装 oracle 10g 软件）

create database（创建数据库）－－startup （open 打开状态）

---------SD(software developer)

－－>登录数据库（建立数据库连接）

－－>SQL语句

echo $ORACLE_SID(环境变量)

oracle db server（oracle 数据库服务器的）的名字

客户端和服务器端程序在同一台机器上 sqlplus

客户端和服务器端程序不在同一台机器上jdbc

（tcp/ip）客户端和服务器端程序在同一台机器上

(D:\> set ORACLE_SID=zhangzenhuan

D:\> set ORACLE_SID

)

%　　setenv ORACLE_SID zhangzenghuan

%　　echo $ORACLE_SID

select语句 select子句 from子句

select投影操作(select子句) 选择操作 连接操作

select子句 列名,表达式,函数

算术表达式　字符表达式(字符串拼接)

empno 员工编号

mgr 该名员工领导的员工编号

语法检查,语义检查,生成执行计划,执行该计划,产生结果集.

select 表达式 列别名 from tabname

字符串 用' '

列别名 用" ", 空格或者大小写敏感

insert 包含null,没值,算术表达式中包含null值,结果一定为null .　null可以按无穷大理解.

nvl(p1,p2) 返回值 nvl(bonus,0) 空值转换函数p1,p2的数据类型一致

pl/sql程序 函数

if p1 is null then

　return p2;

else

　return p1;

end if;

如：select ename,(salary+nvl(bonus,0))*12 "nian XIN" from huan2;

2 一些函数

复制新表

create table newfile

as

select *form oldfile;

---

select（列名，函数，表达式）

投影操作　选择我操作　连接操作

＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊

（字符串表达式－－字符串的拼接）

select ename||','||job　emp from huan2;

字符串的拼接

select ename||"'s total salary is '||salary from huan2;

zhangwuji's total salary is 10000

如若想要输出‘应该输入’“‘两端的表示分号

****

（catch到内存中）（只要有语句执行过那么就相同的语句不必生成执行计划了，减少计算量）

语法检查 语义检查，生成执行计划，执行计划，产生结果集

****

select 表达式 列别名　from　源表

select 表达式　“Zhang　Huan ” from　源表

（“” 中描述的是大小写敏感 或者有空格）

（字符串为单引号）

****

insert 包含了null，没有存值，那么算术表达式的结果一定为null　。null可以按无穷大理解

distinct　bonus ，　bonus包含有多个null，结果集有只有一个null

****

去重复（distinct）

select distinct bonus from huan2;

3 联合去重（当两个都相同时才去除出重复）

select distinct deptno,job from huan2;

distinct 不同 区别 去重

distinct deptno,job from emp_hiloo distinct之后,from之前的所有列联合去重

****

select from where

where子句实现选择操作 where 条件表达式 列名

比较运算符值where salary > 5000

where子句执行在select子句之前,所以列别名不能用于where子句

表达式 比较运算符 值(尽量不用)

between and 闭区间 范围 >= and <=

in (集合) <=> =any(跟集合里的任意一个值相等) <=> = or =

ORA-01797: this operator(运算符) must be followed(跟) by ANY or ALL

c1 between 10 and 20

c1 in (10,20)

select ename,job from huan2　where job='clerk';

一定要加上单引号(不加或者是加双引号就认为是列名)

****

upper（job）

lower（job）

select ename,job from huan2 where upper(job)='CLERK';

查找字符时区分大小写

betweem　and

where salary between　5000 and 1000;

++++++++++++++++++++++++++++++++++

select ename ,salary from huan2 where salary>5000 and salary<10000:

++++++++++++++++++++++++++++++++++

in (集合)<=>ang

必需需跟着ang或者all

where job in('clerk','Manager');

++++++++++++++++++++++++++++++++++++++

clear scr　清屏

like 像　字符

%表示0或者多个字符

_任意一个字符

_%有两个含义　一个代表统配符号，另一个代表他本身

select ename from huan2 where ename like '_a%';

select ename ,job from huan2 where job like 'j_%‘escape'\';

3 关于 null 的讨论

insert 包含null,没值,算术表达式中包含null值,结果一定为null.null可以按无穷大理解.

distinct bonus,bonus包含多个null值,结果集包含一个null值.

判断一个列的取值是否为空,用is null

null = null 不成立 1 <> null 不成立 null <> null 不成立

对于in来说,集合里面是否包含null值,对结果集没影响

对于not in来说,集合里面包含null值,结果集一定是no rows selected.(没有任何记录)

null＝null；不成立　应该使用is null

null<>null　不成立

1<>null　也不成立　<>不等于

is　null　的否定形式是 is not null

＝的否定形式 是<>或者是！＝　（不等于）

between and　的否定形式　not between　and

like　的否定形式　not like

in 的否定形式　not　in　<=>　<>all （和集合里面的所有值都不相等）<=>　<>and<>

对于in来说集合中有无null对结果集没有影响

而对于not　in　来说 集合如果包含null那么结果

集没有任何记录

1 添加 jdbc 驱动

单击右键->build path->Configue build path　之后再选 libraries　选显卡　的add External ja

new java.sql.Date(birthday.getTime());

Oracle（Day2）

函数 function

sysdate　无参数的函数

upper(ename)

ename值参数　函数有返回值,返回值指定数据类型

单行函数　upper(p1)

p1是参数,按参数的数据类型分类

字符函数　upper lower

数值函数　round trunc

日期函数　sysdate add_months

一般函数　nvl(bonus,0)

nvl(ename,'A'),nvl(hiredate,sysdate)

转换函数　to_date,to_char,to_number

多行函数(组函数)

数值函数　round trunc

round(p1,p2)　四舍五入

round(1.235,2)　--> 1.24

round(1.235,0) = round(1.235)　--> 1

round(15.5,-1)　20

trunc(p1,p2)　截取

trunc(1.235,2)　--> 1.23

trunc((15.5,-1)　10

trunc(p1,p2)截取

trunc（1.235,2）--->1.23

trunc(15.5,-1)--->10

＋＋＋＋＋＋＋＋＋＋＋＋＋＋＋＋＋＋＋＋＋＋

日期类型

session　会话　通过connection创建session

alter(修改) session set (设置)

SQL> alter session

set nls_date_format = 'yyyy mm dd hh24:mi:ss';

date　世纪　年　月　日　时　分　秒（固定的7个字节）

世纪年指的是4位数字的年

缺省的日期　DD-MON-RR

ORA-01861: literal(文字值) does not match(匹配)

format string(格式串)

to_date to_char　说明date类型是格式敏感的.

2008 08 08 08:08:08

****

create table test(c1 date) (date后面不要跟宽度，因为它是固定的就是7个字节)　创建的是新表　就如同

create table dept_test(dname varchar(10));

****

insert into test values(to_date ('01-JAN-12','DD-MON-RR'));

insert into test values(to_date ('2012-01-08','yyyy-mm-dd'));

select to_char(hiredate,'yyyy-mm-dd')from emp_huan;

to_date to_char　说明date类型是格式敏感的.

insert　into test values(to_date('2008 08 08 08:08:08','yyyy mm dd hh24:mi:ss'));

select to_char(c1,'yyyy-mm-dd')from test;

1请找出3月份的入职的员工

select ename ,hiredate from emp_huan where to_char(hiredate,'fmmm')='3';

fm　去掉前导

＋＋＋＋＋＋＋＋＋＋＋＋＋＋＋＋＋＋＋＋＋＋＋＋＋＋＋＋

隐式转换　－　>　系统转换

显示转换　用户调用转换函数

'03'=3　缺省方式：字符型－－－>数值型

产生一个日期值　to_date　函数的返回值是日期类型　to_date　　(插入的时候)

处理一个日期值　to_char　处理的参数是日期类型

to_char(hiredate,'mm');　（select的时候）

to_number('03') = 3　　char_to_number

to_number('ab')　提示：invalid(无效) number　转换会异常

to_char(salary)　　number_to_char

to_char(sysdate)　　date_to_char

＋＋＋＋＋＋＋＋＋＋＋＋＋＋＋＋＋＋＋＋＋＋＋＋＋＋＋＋＋＋＋＋＋＋

日期运算

日期 +(-) 数值　日期 - 日期

日期函数

add_months(sysdate ,1);add_months(sysdate,-6);

months_between(sysdate,hiredate)相差了几个月

last_day(sysdate)某月的最后一天

case　when

```
select ename ,salary,
 case when deptno=10    then salary*1.1
      when deptno=20    then salary*1.2
   else
            salary
 end new_sal from emp_huan;
```

没有else的话反回值可能为空

使用范围可能更广这种语法

＊＊＊＊＊＊＊＊＊＊＊decode＊＊＊＊＊＊＊

```
select   ename ,salary ,deptno,
      decode(deptno,10,salary*1.1,
                    20,salary*1.2,
                     salary)new_sal
 from emp_huan;
```

条件不满足的情况下返回值为null

＋＋＋＋＋＋＋＋＋＋＋＋＋＋＋＋＋＋＋＋＋＋＋＋＋＋

排序

order by是最后一个子句, order by在select之后执行

order by　列名　asc(升序,缺省) desc(降序)

order by后面跟列名,列别名,表达式(函数),位置

字符,数值,日期都可以排序

＋＋＋＋＋＋＋

```
SQL> select ename ,bonus
   from emp_huan
    order by bonus desc;
```

null　为无穷大所以在最前面

＋＋＋＋＋＋＋＋＋＋＋

```
select ename ,salary*12 ann_al
     from emp_huan
    order by ann_al   desc；可以根据　列别名
```

＋＋＋＋＋＋＋＋

```
     select ename ,salary*12 ann_al
     from emp_huan
    order by 2   desc；      // 第二列
```

select ename ,deptno,salary*12 ann_sal
from emp_huan
order by deptno, salary desc;

多行函数(组函数)

avg（）平均值　　数值
avg（all　）　avg（distinct　）
sum（）求和　　数值
count（）计数　　数值 字符 日期
count（ *）　返回记录数
min（）　最小值　　数值 字符 日期
max（）　最大值　　数值 字符 日期

组函数处理的是所有的非空值,count(*)返回记录数,空值不影响

当所有的值都是null的时候avg sum max min返回的是null ，count返回的是0

++++++++++++++++++++++

group by

ORA-00937: not a single-group group function不是组函数

在没有group by的情况下,select后面只要有一个组函数,其他的必须是组函数

在有group by的情况下,select后面可以跟组标识(group by后面),组函数

having

from->where->group by->having->select->order by

select　deptno,round(avg(salary))asal
from　emp_huan
group by deptno
order by deptno;

group by如果有多个null，那么多个null放在一起

第三道题

select max(deptno),round(avg(salary)) from emp_huan where deptn=10;

select deptno,round(avg(salary))
from emp_huan
where deptn=10
group by deptno;

在没有group by的情况下，select后面要有一个组函数，其他的必须是组函数

在有group by的情况下select的后面可以跟 组标识

++++++++++++++++++++

各个部门不同职位的平均工资

select　deptno，job，round（avg（salary））
from emp_huan
group by deptno ,job
order by 1

select后出现的列，凡是没有被组函数包围的列，必须出现在group by 短语中 （没有的出现的就会出错）

如果group by短语中的列没有出现在select短语中，不会出错，信息不够全

having子句用于对分组后数据进行过滤

group by 后面的列为组标志，　有group by子句，select后面可以跟组标志和组函数，其他的列名就不可以了。

没有group by子句，select后面只要有一个组函数，其他就都是组函数

前提是结果符合业务逻辑

++++++++++++++++++++++

各公司不同的职位的个数？？

select count(distinct job) from emp_huan

where子句只能过滤记录

where 和 having 的比较

功能都是过滤，都在select之前执行

where过滤的是记录，可以跟任意的列名，可以跟单行函数，不能跟组函数

having过滤的是分组，可以跟组标识，可以跟组函数，不能跟单行函数，不可以跟任意列名

关于 null 值的讨论

case when ,decode在什么情况下返回null.(条件不满足)

asc null在最后 desc null在前面

当所有的值都是null,avg,sum,max,min返回null,count返回0

group by时,如果有多个null值,分为一组

那些职位的人数超过两人
10部门和20部门的工资
> select job
from emp_huan
group by job
having count(ename)>2;

++++++++++++++++++++++

coalesce() 和nvl的用法相同　就是参数列表有三个

++++++++++++++++++++++++++++++++++++

Day03

子查询

非关联子查询

单列子查询

t1表里的列和t2表里的列没有写成条件表达式

t1.c1 = t2.c2

select

from t1

where c1 > (select c2 from t2 ...)

先执行子查询,当返回多条记录时,系统自动去重,应该选择多值运算符如in.

=表达single-row subquery 单行子查询

ORA-01427: single-row subquery returns more than one row

单行子查询返回多行

子查询

非关联子查询

多列子查询

哪些员工的工资等于本部门的平均工资

where (deptno,salary) in (select deptno,round(avg(salary))...)

子查询

关联子查询

t1表里的列和t2表里的列写成where条件表达式

查询从主表中依次取得记录,子查询根据主表的记录执行.因此子查询会执行多遍.

注意： 组函数是忽略空值的

先执行子查询，当返回多条记录时，系统会［自动去重］，

应该选择多值运算符（如：in）。不需要在子查询里写distinct子句

not in　　<>all

哪些员工的工资等于本部门的平均工资

select　ename ,deptno ,salary ,from emp_huan

where (deptno ,salary)in (select deptno ,round (avg(salary))

from emp_huan

group by deptno)

关联子查询

(子查询不在是独立的sql语句，需要依赖主查询传来的参数，这种方式叫关联子查询)

那些员工的工资大于本部门的平均工资

select ename ,deptno ,salary from emp_huan n

where salary>(select round(avg(salary))

from emp_huan i

where i.deptmo=n.deptno)

－－ 子查询不在是独立的sql语句， 需要依赖主查询传来的参数n.deptno

先进行主查询，然后在把参数传给子函数 效率很低

++

集合操作

用集合操作运算符将多个select语句连接起来，将结果集看成集合

要求select语句是同结构的，列的个数以及数据类型一致

并集

union（去重记录）　union all（不去重复记录）

intersect（交集　去重记录）

minus（差　去重记录）

将以下写法改为case when

select ename,deptno,salary*1.1

from emp_hiloo

where deptno = 10

union all

select ename,deptno,salary*1.2

from emp_hiloo

where deptno = 20

union all

select ename,deptno,salary

from emp_hiloo

where deptno not in (10,20)

交集

intersect(去重记录)

select deptno from dept_hiloo

intersect

select deptno from emp_hiloo

差

minus(去重记录),select的顺序改变了结果集不一样

select deptno from dept_hiloo

minus

select deptno from emp_hiloo

****

多表查询（表连接）　三种连接方式有三种生成结果集的方式

cross join（交叉连接） 笛卡尔积

从t1表中任取一条记录和t2表中任取一条记录组合放入结果集,最后的个数为m*n,m为t1的记录数,n为t2的记录数

inner join（内连接） 驱动表　匹配表

两张表的任意一条记录要想出现在结果集中,必须在另一张表中根据on条件找到匹配的记录.如果解决的是匹配问题,用inner join实现.

on e.deptno = d.deptno 等值连接

ORA-00905: missing(丢失) keyword(关键字)

谁做驱动表不影响结果集

outer join（外连接）

left表示左边的表必须做驱动表

right表示右边的表必须做驱动表

驱动表里的记录在结果集里一个都不能少

from t1 left join t2

on t1.c1. = t2.c2

外连接结果集=内连接的结果集 ＋t1表中匹配不上的记录和t2表中的一条null记录的组合

from t1 right join t2

on t1.c1. = t2.c2

外连接结果集=内连接的结果集 ＋t2表中匹配不上的记录和t1表中的一条null记录的组合

from t1 full join t2

on t1.c1. = t2.c2

外连接结果集=内连接的结果集 ＋t1表中匹配不上的记录和t2表中的一条null记录的组合+ t2表中匹配不上的记录和t1表中的一条null记录的组合

什么情况下用外连接?

1 员工名称 领导名称,包括zhangsanfeng,除了匹配的记录,匹配不上的记录也要出现在结果集中.

2 哪些人不是领导? 否定问题,结果集中只要匹配不上的记录

outer join + where 匹配表.pk列 is null <=> not exists not in

子查询in not in

子查询 exists not exists

表连接 inner join outer join 10=9+1 15=9+6

以上查询形式,结果集是如何产生的.

****

outer join（外连接）最重要的是谁做驱动表（一个都不能出现在结果集中）

left左边的驱动表

from t1 left join t2 on t1.c1=t2.c2

from t1 right join t2 on t1.c1=t2.c2

from t1 full join t2 on t1.c1=t2.c2

t2没有的匹配的记录，就会全部填空外连接是一个都不能少，都要出现在结果集里。

外连接的结果集＝内连接的结果集＋t1表中匹配不上的记录和t2中的一条null记录组合

张三风 找到 员工 领导

select e.ename,nvl(m.ename,'BOSS')

from emp_huan e left join emp_huan m

on m.empno=e.mgr

****

select ename from emp_huan

where empno in(select mgr from emp_huan

where ename='zhangwuji');

张wuji 领导谁

select ename from emp_huan

where mgr in(select empno from emp_huan

where ename='zhangwuji');

****

Oracle（Day04）

内连接

and where 都是允许，先过滤后连接

外连接分为两种情况

and在连接之前，先对匹配表进行过滤，用关键字and

where在外连接之后，需要通过匹配表的列对外连接结果过滤时，用关键字where

对驱动表的过滤必须用关键字where

1 哪些部门没有员工zhangwuji

from emp_huan e join dept_huan d

on e.deptno=d.deptno

and e.deptno='zhangwuji'

where e.empno is null;

select d.dname

from

连接形式　:等值连接　非等值连接（between　and）

自连接

连接

从结果集的产生分类

cross join

inner join

　　等值连接(两张不同表,描述共同属性的列)

　　非等值连接 (between and)

　自连接(同一张表的不同列有关系)

outer jion

　　等值连接(两张不同表,描述共同属性的列)

　　非等值连接 (between and)

　　自连接(同一张表的不同列有关系)

DML

insert into　values 只能插入一条记录

insert into tabname values ('a','b',null,1) 顺序(desc tabname)

insert into tabname(c1,c2) values ('a','b')

insert into tabname select 插入多条记录

++

rownum 伪列 记录号,行号

rownum必须从1开始才能选出记录.

分页问题

第一页 where rownum <= 10

第二页

列出第3条至第6条记录

select rn,ename,salary

from (select rownum rn,ename,salary

　　from emp_hiloo

　　where rownum <= 6)

where rn >= 3

排名问题

工资最高 的前3名员工?

select rownum,ename,salary

from (select ename,salary

　　from emp_hiloo

　　order by salary desc)

where rownum <= 3

update

where 同select,确定哪些记录应该修改

set colname = value

set bonus = null　=表示赋值

where bonus = null　=表示相等 false

delete删除一条记录

delete from tabname

where 同 select 确定哪些记录应该删除

****

子查询

1　非关联子查询（主子查询得到表之间没有建立关联关系）先执行子查询，返回结果给主查询

　主查询会对 子查询的返回结果 重复的 去重

　返回的结果是多值的时候　要用多值运算符**不能用单值运算符**

5 用sql脚本实现转帐脚本，创建一张帐户表，两列（帐号，余额）

```
drop table huan_;
create table huan_(count char(5),yu number(10));
insert into huan_ values('a',4000);
insert into　huan_ values('b',100);
commit;
update huan_　yu=yu-100
where count='a';　不太完全????
```

date

char　有缺省宽度为1　，按定义长度存，取值若固定用char，对空格不敏感

varchar2 必须定义宽度，按实际长度存，取值不固定，用varchar2

```
select ename ,hiredate
from emp_huan
where to_char(hiredate,'fmMONTH)='MARCH';//应该去掉空格
```

返回的结果是varchar2类型的

TCl

transaction 事务

事务结束 commit(提交,入库)　rollback(回滚,取消操作)

DML 把旧数据放入rollback segment(回滚段 公共资源),

当你commit或者rollback,即事务结束回滚段里的数据释放.

commit/rollback

DMLS

commit/rollback

DDL是自动提交的

事务的并发操作,操作同一张表,定义事务的隔离级别,缺省事务的隔离级别read committed.读已经提交了的数据和本session正在处理的数据.

DML加锁方式

加锁粒度(级别) 表 行

锁性质　共享 排他

表级共享锁

行级排他锁

	表级共享锁	行级排他锁
s1	ok	ok
s2	ok	wait
s3	ok	ok

DDL加锁方式 DDL排他锁

ORA-00054: resource busy(资源忙,表忙,DML操作表) and acquire获得 with NOWAIT不等待 specified 指定

为什么要commit,rollback

1 你操作的数据其他session看不见

2 你加在表上的锁不释放,会阻塞其他人的操作

3 你占用的回滚段资源不释放

Oracle（day05）

constraint 约束

primary key pk 主键

foreign　key fk 外键

not null 非空

unique 唯一键

check 检查

约束的定义形式

列级约束： 列名　数据类型 constraint 约束名　约束类型

1 pk = not null + unique 唯一且非空 一张表里只能有一个pk约束

保证表里不会出现两条完全一样的记录

ORA-00001: unique(唯一) constraint (HILOO.TEST_C1_PK) violated(冲突)

ORA-00001: unique constraint (HILOO.SYS_C0079936系统起的约束名字) violated

表级约束

列名 数据类型,

constraint 约束名 约束类型(列名)

2 not null 没有表级约束形式

c1 number not null,

ORA-01400: cannot insert NULL into ("HILOO"."TEST"."C1")

3 unique

c1 number constriant test_c1_uk unique,

c1 number,

c2 numbre,

constraint test_c1_uk unique(c1)

uk允许为空,而且是多个空值

pk与uk区别

pk not null,uk null

pk 1 table只能有1 pk,uk 可以多个uk

pk,uk都要求唯一

4 check

列级约束

create table test(

c1 number constraint test_c1_ck check (c1 > 100))

表级约束

c1 number,

constraint test_c1_ck check (c1 > 100))

5 FK

实现两张表的多对一关系,多定义成fk,一定义pk/uk.

一个员工属于一个部门,一个部门里有多个员工

在child table的某列定义Fk,它的取值要引用parent table对某列,该列要求唯一特性(pk/uk)

1 先create parent table(pk/uk),再create chlid table(fk列)-->parent(pk 列)

2先insert into parent table,再insert into child table

3 先delete from chlid table,再delete from parent table

4先drop child table,再drop parent table

create child table

ORA-02270: no matching(匹配) unique or primary key for this column-list(列列表)

```
SQL> alter table parent
  2  add constraint parent_c1_pk primary key(c1);
```

```
SQL> l
  1  create table child(
  2  c1 number constraint child_c1_pk primary key,
  3  c2 number(3) constraint child_c2_fk
  4*                 references parent(c1))
```

insert into child table

ORA-02291: integrity(完整性) constraint (HILOO.CHILD_C2_FK) violated(冲突) - parent key not found(父键值没发现)

delete from parent

ORA-02292: integrity constraint (HILOO.CHILD_C2_FK) violated - child record found(发现了子记录)

drop parent table

ORA-02449: unique/primary keys in table referenced(引用) by foreign keys(外键)

drop table parent cascade(级联) constraints purge;

先将child table上的fk约束drop,断绝父子关系

on delete cascade

```
create table child1(
c1 number(2) constraint child1_c1_pk primary key,
c2 number(3) constraint child1_c2_fk
                references parent(c1)
                on delete cascade)
```

on delete cascade (级联删除) 影响对parent table的delete操作,

系统先delete from child,再delete parent table.

```
create table child2
(c1 number(2) constraint child2_c1_pk primary key,
 c2 number(3) constraint child2_c2_fk
                references parent(c1)
                on delete set null)
```

on delete set null(删除之前设置为空)影响对parent table的delete操作.

先将child 表的c2列=1的值update成null,在delete from parent 中c1=1的记录

Fk的三种定义形式

```
c2 number(3) constraint child2_c2_fk references
parent(c1)
c2 number(3) constraint child2_c2_fk references
parent(c1)
                                on delete cascade
c2 number(3) constraint child2_c2_fk references
parent(c1)
                                on delete set null
```

表级约束

```
c2 number(3),
constraint child_c2_fk foreign key(c2) references
parent(c1)
```

3%%%%%%%%%%%%%%%%%%%%%%%%%%%%

3三张表

```
student
        id      pk
        name    not null
course
        id      pk
        name not null
stu_cour
        sid     fk ---> student(id)
        cid     fk ---> course(id)
        pk(sid,cid)
        score   check   [0,100]
```

database object

table

view

index

sequence

function

视图

view是一条select语句,不占存储空间,通过view查询,还是查源表

```
SQL> select text from user_views
  2  where view_name = 'TEST_V1';
```

获得view的定义

SQL> select object_name,object_type,status

2　from user_objects

3　where object_name = 'TEST_V1';

获得view的状态　valid(有效) invalid(无效)

alter view test_v1 compile;

编译view,让状态invalid -->valid

view的作用

1 简化查询

2 安全,限制用户查看的数据(子集),定义view,grant view给用户,而不是table

3 table的并集

create view　北京　as

select from　海淀

union all

select　from　朝阳

union all

select from　昌平

select * from　北京

view分类

简单view select from　一张表,没有运算　能DML

复杂view　各个部门的平均工资,部门名称,平均工资

create or replace view d_avgsal_v

as

select max(d.dname) dname ,round(avg(e.salary)) avgsal

from emp_hiloo e join dept_hiloo d

on e.deptno = d.deptno

group by d.deptno

ORA-00998: must name this expression(表达式) with a column alias(列别名)

view的约束

read only view

create or replace view test_ro

as

select * from test

with read only;　--约束

不能对只读视图做DML操作.

ORA-01733: virtual column(虚拟列) not allowed here

check view

create or replace view test_ck

as

select * from test

where c1 = 1;

with check option --约束

drop view viewname

sequence

序列号,产生唯一值

s1.nextval　返回一个新的唯一值

s1.currval　返回当前session的取值是多少,必须先执行s1.nextval,s1.currval才能返回值

drop sequence seqname

student表的id的取值用sequence实现

course表的id的取值用sequence实现

stu_cour (sid,cid)

insert into student values (stu_seq.nextval,...)

insert into course values (cour_seq.nextval,...)

insert into stu_cour values (1001,101....)

delete from tabname

truncate table tabname

把表里所有记录删除

区别

1 delete不动HWM,不释放空间;truncate table移动HWM,它之下的数据仅有有限的几个,释放空间

2 delete占rollback segment空间,慢;truncate table不占rollback segment空间,快

3 delete数据可恢复,truncate table数据不可恢复

不适合用delete删一张大表.

索引

index

记录存储在数据块(data block)中,data block是做I/O的最基本单位

HWM hign water mark,高水位线,曾经插过记录的最大数据块的位置

FTS full table scan,全表扫描,把HWM以下的所有data block全部读一遍

select ename,salary from emp_hiloo

where empno = 1001;

create database　创建一堆文件.数据文件,日志文件,控制文件

ROWID 代表一条记录的物理位置

data_object_id

一条记录是属于哪张表的

datafile_id

一条记录是属于哪个数据文件的

datablock_id

一条记录是属于数据文件里的哪个数据块的

row_id

一条记录是数据块里的第几条记录

基于索引的扫描

index里记录rowid

create index test_c1_idx on test(c1);

给表test的c1列建索引

index的结构

平衡树,根节点,分支节点,叶子节点

叶子节点存储索引项(index entry),由key,rowid组成,key值是某条记录的c1列的取值,rowid是该记录的物理位置.所有的叶子节点串成了双向列表.

根节点,分支节点是用来导航的,里面存储下一个节点的物理位置和该节点里存储的数据范围.

索引扫描快于FTS

降低了扫描data block的数量,从硬盘上读phisical read(物理读),从内存里读,data buffer gets(logical read)逻辑读.

付出代价:空间代价,DML变慢

结果集里的数据少,数据表的数据大

哪些列适合建索引

1经常出现在where子句中的列

2经常用于表连接的列

3 主键列,唯一键列(系统会自动键唯一性索引)

create unique index test_c1_idx

on test(c1);

在一个列上定义唯一性约束和唯一性索引是等价的.

4 外键列　on p.pk column = c.fk column

　　on e.deptno = d.deptno

emp表上empno(pk),mgr(fk)

on e.mgr = m.empno

5 order by列,group by　列,distinct列

6 where c1 is null,一定是FTS,索引里根本不计null值

7 列里有大量null值,找not null值会快

哪些写法会导致索引用不了

where salary*12 > 60000

where upper(ename) = 'LISA'

where c1 = 1 c1 is varchar2

如果在这些列上有索引,索引用不上

函数索引

create index test_c1_idx on test(round(c1));

student name index

course name index

stu_cour score index

drop index test_c1_idx

占空间 DML慢

SQL

table view index sequence

PL/SQL procedure language/SQL

编辑,编译函数

create or replace function f_avgsal

(P_deptno number)

p_deptno 参数名

return number 定义函数的返回类型

is

　v_avgsal　emp_hiloo.salary%type;

　把变量v_avgsal定义成跟emp_hiloo表的salary列的数据类型一样

begin

　select round(avg(salary)) into v_avgsal

　from emp_hiloo

　where deptno = p_deptno;

　把查询语句的结果赋给变量v_avgsal

　return v_avgsal;　定义函数的返回值

end;

/

SQL> select f_avgsal(10) from dual; --调用函数

warning: Function created with compilation(编译)

errors.

show error

update account

set bal = (case when id = 'A' then bal - 1500

　　when id = 'B' then bal + 1500

　end)

where id in ('A','B')

++++++++++++++++++++++++++++++++

课堂练习

课堂练习

1列出员工名称以及工资

select ename,salary from emp_hiloo

2 列出员工名称以及年薪

select ename,salary*12　from emp_hiloo

3列出员工名称以及一年的总收入

select ename,(salary+nvl(bonus,0))*12 tot_sal from emp_hiloo

4公司里有哪些不同的职位

select distinct job from emp_hiloo

5公司里每个部门有哪些不同的职位

select distinct distinct deptno,job from emp_hiloo

6工资大于5000的员工的名称和工资

select ename,salary from emp_hiloo where salary > 5000

7工资大于5000的员工的名称和年薪

select ename,salary*12 from emp_hiloo where salary > 5000

8年薪大于60000的员工的名称和年薪

select ename,salary*12 from emp_hiloo where salary > 5000

9哪些员工的职位是clerk

select ename,job from emp_hiloo where job = 'clerk'

10 哪些员工的职位是Manager

select ename,job from emp_hiloo where job = 'Manager'

11 哪些员工的职位是clerk,不知道clerk的大小写

select ename,job from emp_hiloo where lower(job) = 'clerk'

12 找出工资在5000到10000之间的员工的名称和年薪

select ename,salary*12 from emp_hiloo

where salary between　5000 and　10000

13 哪些员工的职位是clerk或Manager或salesman

select ename,job from emp_hiloo

where job in ('clerk','Manager','salesman')

14 哪些员工的名字的第二个字符是a.

select ename,job from emp_hiloo

where ename like '_a%'

15哪些员工的职位的前两位字符是j_ (j_salesman符合条件)

转义

select ename,job from emp_hiloo

where job like 'j_%' escape '\'

16 哪些员工没有奖金

select ename,bonus from emp_hiloo

where bonus is null

17 哪些员工有奖金

select ename,bonus from emp_hiloo

where bonus is not null

哪些员工没有工资

where bonus is null ;

++++++++++++++++++++++++++++

课堂练习

1 请找出3月份入职的员工.

```
SQL> select ename,hiredate
  2  from emp_hiloo
  3  where to_char(hiredate,'mm') = '03';
SQL> l
  1  select ename,hiredate
  2  from emp_hiloo
  3* where to_char(hiredate,'fmmm') = '3'
```

fm 去掉前导0或者两端的空格

2 十分钟之后

```
SQL> select sysdate,sysdate+1/144
  2  from dual;
```

3 列出ename,empno,mgr,其中zhangsanfeng的mgr显示Boss

4 case when和decode

```
  1  select ename,salary,deptno,
  2         case when deptno = 10 then salary*1.1
  3              when deptno = 20 then salary*1.2
  4         else
  5              salary
  6         end new_sal
  7* from emp_hiloo

SQL> select ename,salary,deptno,
  2         decode(deptno,10,salary*1.1,
  3                        20,salary*1.2,
  4                        salary) new_sal
  5  from emp_hiloo;
```

5 列出员工名称和年薪,按年薪降序排列

```
select ename,salary*12
from emp_hiloo
order by salary desc
```

6 列出员工名称,部门号和年薪,按部门号升序,年薪降序排列

```
SQL> select ename,deptno,salary*12 ann_sal
  2  from emp_hiloo
  3  order by deptno , salary desc
```

7 求奖金的平均值,和,最小值,最大值,个数(所有有奖金的)

```
select avg(bonus),sum(bonus),min(bonus),max(bonus),
```

```
        count(bonus)
from emp_hiloo
```

8 各个部门的平均工资,部门号,平均工资

```
select deptno,round(avg(salary))
from emp_hiloo
group by deptno
```

9 列出来每种奖金有多少个人

```
select bonus,count(empno)
from emp_hiloo
group by bonus
```

10 列出10部门的平均工资,只要平均工资

```
select round(avg(salary))
from emp_hioo
where deptno = 10
```

11 列出10部门的平均工资,部门号,平均工资

```
 1  select deptno,round(avg(salary))
 2  from emp_hiloo
 3  where deptno = 10
 4* group by deptno

select max(deptno),round(avg(salary))
from emp_hioo
where deptno = 10
```

12 各个部门不同职位的平均工资?

```
select deptno,job,round(avg(salary))
from emp_hiloo
group by deptno,job
order by 1
```

13 该公司不同的职位的个数?

```
 select count(distinct job) from emp_hiloo;
```

14 该公司不同奖金的个数?

```
select count(distinct nvl(bonus,0))
from emp_hiloo
```

15 在10,20部门中,哪些部门的平均工资高于5000?,按部门号升序排列

```
select deptno,round(avg(salary))
from emp_hiloo
where deptno in (10,20)
group by deptno
having round(avg(salary)) > 5000
order by deptno
```

16 列出10,20部门的平均工资

```
select deptno,avg(salary)
from emp_hiloo
where deptno in (10,20)
group by deptno

select deptno,avg(salary)
from emp_hiloo
group by deptno
having deptno in (10,20)
```

16 哪个员工的工资是最低的

```
select ename,salary from emp_hiloo
where salary = (select min(salary) from emp_hiloo)
```

课后练习

1 <5000 涨10%　[5000,10000] 涨 5% 其他人不动

```
select ename,salary,
        case when salary < 5000 then salary*1.1
             when salary between 5000 and 10000
                                        then
salary*1.05
             else
                  salary
        end new_sal
from emp_hiloo
```

2 哪些部门的平均工资比30部门的平均工资高

```
select deptno,round(avg(salary))
from emp_hiloo
group by deptno
having round(avg(salary)) > (select round(avg(salary))
                                    from emp_hiloo
                                    where deptno =
30)
```

++++++++++++++

课堂练习

1 哪些人是领导?

```
select ename from emp_hiloo
where empno in (select mgr from emp_hiloo)
in      =any
4 rows selected
```

2 哪些人是员工?不是领导?

```
select ename from emp_hiloo
where empno not in (select mgr from emp_hiloo
                          where mgr is not null)
not in      <>all
```

3 哪些员工的工资等于本部门的平均工资

```
select ename,deptno,salary
from emp_hiloo
where (deptno,salary) in (
                          select
deptno,round(avg(salary))
                          from emp_hiloo
                          group by deptno)
```

3 哪些员工的工资大于本部门的平均工资

```
select ename,deptno,salary
from emp_hiloo o
where salary > (select round(avg(salary))
                    from emp_hiloo i
                    where i.deptno = o.deptno)
```

4 哪些人是领导?

```
select ename from emp_hiloo o
where exists (
                select 1 from emp_hiloo i
                where o.empno = i.mgr)
```

5哪些部门有员工,列出部门名称

```
select dname from dept_hiloo o
where exists
            (select 1 from emp_hiloo i
             where o.deptno = i.deptno)

select dname from dept_hiloo o
where deptno in (select deptno from emp_hiloo)
```

6哪些部门没有员工

```
select dname from dept_hiloo o
where not exists
            (select 1 from emp_hiloo i
             where o.deptno = i.deptno)
select dname from dept_hiloo o
where deptno not in (select deptno from emp_hiloo)
```

7 哪些人是员工,哪些人不是领导?

```
select ename from emp_hiloo o
where not exists
              (select 1 from emp_hiloo i
               where o.empno = i.mgr)
```

8 列出员工的名称和部门名称

```
select e.ename,d.dname
from emp_hiloo e   join   dept_hiloo d
on e.deptno = d.deptno
```

9 列出beijing地区有哪些员工?

```
select e.ename,d.dname
from emp_hiloo e   join   dept_hiloo d
on e.deptno = d.deptno
and d.location = 'beijing'
```

10 zhangwuji在哪个地区上班?

```
select e.ename,d.location
from emp_hiloo e    join    dept_hiloo d
on e.deptno = d.deptno
and e.ename = 'zhangwuji'
```

11 列出员工的名称以及他的领导的名称

不包含zhangsanfeng

```
select e.ename,m.ename
from emp_hiloo e join emp_hiloo m
on e.mgr = m.empno
```

e 10 9 1 (匹配9 zhangsanfeng 不匹配 1)

m 10 4 6 (匹配4条 6条不匹配)

包含zhangsanfeng

```
select e.ename,m.ename
from emp_hiloo e join emp_hiloo m
on e.mgr = m.empno
union all
select ename,'Boss'
from emp_hiloo
where mgr is null
```

```
select e.ename employee,
        decode(m.ename,e.ename,'Boss',
                        m.ename) manager
from emp_hiloo e join emp_hiloo m
on nvl(e.mgr,e.empno) = m.empno
```

e 10 10 (匹配10)

m 10 4 6 (匹配4条 6条不匹配)

```
select e.ename employee,
        nvl(m.ename,'Boss') manager
from emp_hiloo e left join emp_hiloo m
on e.mgr = m.empno
```

10=9+1

12 哪些人是领导?(用inner join)

```
select distinct    m.ename
from emp_hiloo e join emp_hiloo m
on e.mgr = m.empno
```

13 哪些人是员工,哪些人不是领导?(用outer join)

```
select m.ename
from emp_hiloo e right join emp_hiloo m
on e.mgr = m.empno
where e.empno is null
```

先执行外连接再执行where过滤,通过匹配表的列过滤外连接的结果集

课外练习

1 zhangwuji的领导是谁? (用 in)

2 zhangwuji领导谁?(用 in)

3 哪些员工的工资比同职位的平均工资高?

4 zhangwuji的领导是谁? (用 inner join)

5 zhangwuji领导谁?(用 inner join)

6 各个部门的平均工资,列出部门名称,平均工资

7 哪些部门没有员工(用 outer join 用 not in)

12 哪些人是领导（用inner join）

13哪些人是员工，那些人不是领导（用 outer join）

＋＋＋＋＋＋＋＋＋＋＋＋＋＋＋

cross join（交叉连接） 很少去写cross join

哪些员工在北京上班

```
select    e.ename ,d.location
from emp_huan e    cross join dept_huan d ;
```

四十条记录

＋＋＋＋＋＋＋＋＋ inner join

哪些员工在北京上班 (inner 可以省略) －－ >驱动表 匹配表 结果集一样

```
select    e.ename , e.deptno, d.location,d.deptno
from emp_huan e inner join dept_huan d
on d.deptno=e.deptno
and d.location='beijing';      //应该现过滤在连接
```

任意一条的表的记录要想出现在 结果集必须要找到匹配的记录

核心是解决匹配问题，如果结果集要想把匹配的记录找出来，需要用内连接

＋＋＋＋＋＋＋＋＋＋＋＋＋＋＋＋＋＋＋＋＋＋＋

5 zhangwuji在哪里上班？

```
select e.ename,d.location
from dept_huan d join emp_huan e
 on d.deptno=e.deptno
//等值连接
and e.ename='zhangwuji';
```

分清 过滤条件，

＋＋＋＋＋＋＋＋

找员工和他的领导 来自同一张表

```
select
e.ename ,decode(p.ename,e.ename,'BOSS',p.ename)
from emp_huan e join emp_huan p
on nvl(e.mgr,e.empno)=p.empno;
```

（自连接self join）同一张表有关系，通过起别名变成两张表

起 别名变成两张表

少张三风 因为mgr为空

＋＋＋＋＋＋＋＋＋＋＋＋＋＋＋＋＋＋＋＋＋＋＋＋＋＋＋＋＋＋＋＋＋＋

列出员工的名称以及他的领导的名称

课外联练习

1 zhangwuji的领导是谁（用 in）

2 zhangwuji领导谁？（in）

3 zhangwuji的领导是谁（用 inner join）

领导是谁

```
select distinct m.ename
from emp_huan e join emp_huan m
on m.empno=e.mgr;
```

1 zhangwuji的领导是谁（用 in）

```
select ename from emp_huan
        where empno in(select mgr from
emp_huan
                where
ename='zhangwuji' )
```

2 zhangwuji领导谁？（in）

```
select ename from emp_huan
    where mgr in(select empno from emp_huan
            where ename='zhangwuji');
```

4 zhangwuji的领导是谁（用 inner join）

```
select e.ename ,m.ename
   from emp_huan e join emp_huan m
on e.mgr=m.empno
and e.ename='zhangwuji';
```

5 zhangwuji领导谁？（inner join）

```
select e.ename ,m.ename
from emp_huan e join emp_huan m
on e.mgr=m.empno
and m.ename='zhangwuji';
```

3 哪些员工的工资比同职位的平均工资高？

```
select e.ename ,e.salary
from emp_huan e
where salary>(select round(avg(salary))from emp_huan
i
                where
```

e.job=i.job);

做了10遍 ，比较慢

++++++++++++++

需要完善

要重视　非常重要

select e.ename ,e.job ,e.salary, a.savg

from emp_huan e join (select job ,round

(avg(salary))savg

from emp_huan

group by job)a

on e.job=a.job

and e.salary>a.savg

@@@@@@@@@@@@@@@@@@@@@@

6 各个部门的平均工资，列出部门名称，平均工资

select d.dname ,round(avg(e.salary))

from emp_huan e join dept_huan d

on e.deptno=d.deptno

group by d.dname;

select max(d.dname) ,round(avg(e.salary))

from emp_huan e join dept_huan d

on e.deptno=d.deptno

group by d.deptno;

如果group by e.deptno;要改　max(d.dname)　　X X X X X

有group by子句select后面跟的要书组标识或者组函数

group by e.deptno;

select d.dname,a.savg

from dept_huan d join

(select deptno,round(avg(salary))savg

from emp_huan

group by deptno)a

on a.deptno=d.deptno

＋＋＋＋＋＋＋＋＋＋＋＋＋＋＋＋＋＋＋＋＋＋＋

8　　哪些人是领导

select ename from emp_huan

where empno in(select mgr from emp_huan)

select distinct m.ename from emp_huan e join

emp_huan m

on e.mgr=m.empno

exist把符合的记录留下，放在结果集中（如果有exists那么子查询的select后不要写组函数，组函数返回空记录）

select ename from emp_huan o

where exists(

select 1 from emp_huan i

where o.empno=i.mgr)

++

2哪些人是员工？（不是领导?）

select ename from emp_huan

where empno not in(select mgr from emp_huan

where mgr is not null)select m.ename

from emp_huan e right join emp_huan m

on e.mgr=m.empno

where e.empno is null

++++++++++++++++++

select ename from emp_huan o

where not exists(select 1 from emp_huan e

where o.empno=e.mgr);

++

++++++++++++++++++++++

7　哪个部门没有员工　　（not exists 是把符合的记录过滤掉）

select dname from dept_huan o

where not exists(

select 1 from emp_huan　i

where o.deptno=i.deptno)

7 哪些部门没有员工 （用outer join 用not in）

select dname from　dept_huan

where deptno not in(select deptno from emp_huan);

select d.dname

from emp_huan e right join dept_huan d

on e.deptno=d.deptno

where e.empno is null

1 表 c1,c2,c3,c1,c2做成联合主键

create table test(

c1 number(2),

c2 number(3),

constraint test_c1_c2_pk primary key(c1,c2),

c3 varchar2(3))

2 表 c1,c2,c3,c4,c5 c1,c2,c3要求唯一且非空,c4,c5联合唯一.

create table test(

c1 number constraint test_c1_pk primary key,

c2 number not null constraint test_c2_uk unique,

c3 number not null constraint test_c3_uk unique,

c4 number,

c5 number,

constraint test_c4_c5 unique(c4,c5))

++

create table

课堂练习

1 哪些部门没有员工.zhangwuji

select d.dname

from emp_hiloo e right join dept_hiloo d

on e.deptno = d.deptno

and e.ename = 'zhangwuji'

where e.empno is null

select d.dname

from (select deptno,ename from emp_hiloo

where ename = 'zhangwuji')　e right join dept_hiloo d

on e.deptno = d.deptno

where e.empno is null

2 列出哪些员工的工资是3,5级

select e.ename,e.salary,s.grade

from emp_hiloo e join salgrade_hiloo s

on e.salary between s.lowsal and s.hisal

and s.grade in (3,5)

3 列出各个级别的人数(不包含0级)

select s.grade,count(s.grade)

from emp_hiloo e join salgrade_hiloo s

on e.salary between s.lowsal and s.hisal

group by s.grade

4 列出各个级别的人数(包含0级)

select s.grade,count(e.empno)

from emp_hiloo e right join salgrade_hiloo s

on e.salary between s.lowsal and s.hisal

group by s.grade

5 用sql脚本,实现转账功能,创建一张账户表,两列(账

号,余额),插入两条记录'a' 4000,'b' 100,完成转账,从账户'a'转1500元钱到账户'b'.

编写和运行sql脚步

1 在linux写完脚步文件,全选,拷贝,在26上用vi test.sql,按i,粘贴,按ESC,按:wq!,脚步文件里是sql语句,每条语句后有;

2 必须在test.sql的目录下执行,

%sqlplus openlab/open123 @test.sql,

或者在文件名前加上所在路径

sqlplus openlab/open123 @../test.sql

63月份入职的员工

```
SQL> select ename,hiredate
  2  from emp_hiloo
  3  where to_char(hiredate,'fmMONTH') =
'MARCH';
```

课外练习

1 zhangwuji的领导是谁? (用 in)

```
select ename from emp_hiloo
where empno in (select mgr from emp_hiloo
                where ename = 'zhangwuji')
```

2 zhangwuji领导谁?(用 in)

```
select ename from emp_hiloo
where mgr in (select empno from emp_hiloo
              where ename = 'zhangwuji')
```

3 哪些员工的工资比同职位的平均工资高?

```
select e.ename,e.salary
from emp_hiloo o
where salary > ( select round(avg(salary)) from
emp_hiloo i
                where o.job = i.job)

select e.ename,e.job,e.salary,a.savg
from emp_hiloo e  join
        (select job,round(avg(salary)) savg
          from emp_hiloo
          group by job) a
on e.job = a.job
and e.salary > a.savg
```

4 zhangwuji的领导是谁? (用 inner join)

```
select e.ename,m.ename
from emp_hiloo e join emp_hiloo m
on e.mgr = m.empno
and e.ename = 'zhangwuji'
```

5 zhangwuji领导谁?(用 inner join)

```
select e.ename,m.ename
from emp_hiloo e join emp_hiloo m
on e.mgr = m.empno
and m.ename = 'zhangwuji'
```

6 各个部门的平均工资,列出部门名称,平均工资

```
select d.dname,round(avg(e.salary))
from emp_hiloo e join dept_hiloo d
on e.deptno = d.deptno
group by d.dname ---
select max(d.dname),round(avg(e.salary))
from emp_hiloo e join dept_hiloo d
on e.deptno = d.deptno
group by d.deptno

select d.dname,a.savg
from dept_hiloo d join
            (select deptno,round(avg(salary)) savg
              from emp_hiloo
              group by deptno) a
on a.deptno = d.deptno
```

7 哪些部门没有员工(用 outer join 用 not in)

```
select dname from dept_hiloo
where deptno not in (select deptno from emp_hiloo)
select d.dname
from emp_hiloo e right join dept_hiloo d
on e.deptno = d.deptno
where e.empno is null
```

课后练习

1 列出按工资排名的第3名至第6名员工?

```
c1 number,
 c2 number(6),
 c3 number(4,2),99.99
 c4 number(2,4)  最大值 0.0099 ,只有后面的添加数字
 c5 number(4,-2)       1250－－1300


 show user
 connect openlab /open123

 删除表
 drop table test purge;
```

新添加的知识!

```
select id ,id*12 id_all from user_name;
```

	ID	ID_ALL
1	1	12
2	2	24

对于相应的字段可以进行(加减乘除)

注意：空值和任何数据做算数运算，结果为空(null)

■ 下列写法得不到正确的结果：

```
SQL>  select ename , salary , bonus , salary+bonus  month_sal
        from emp_xxx ;     --错误的写法
```

2. 处理空值的函数 nvl，使用方式： nvl(bonus, 0)

■ 正确写法：如果 bunus 的值是 null，则取 0

```
SQL>  select ename, salary , bonus , salary + nvl(bonus, 0) month_sal
        from emp_xxx ;
```

小例子:

```
select id, nvl(id,0)+10 id_add from user_name;
```

当插入空值时要用 null 表示

```
insert into user_name
values (null,'xian','test')
```

考察的是||的用法.

```
select id||'连接字符 '||name  id_name  from user_name;
```

	ID_NAME
1	1连接字符 admi
2	2连接字符 hu
3	连接字符 add

如果想要出现' 应该用''

如何复制表

```
create table user_name2 as select * from user_name;
```

---

distinct 的具体用法

4.5. distinct 关键字

注意：distinct 必须(只能)跟在 select 后边

【案例 10】机构中有多少种职位？

```
SQL>  select distinct job from emp_xxx ;
```

【案例 11】员工分布在哪些部门？

```
SQL>  select distinct deptno from emp_xxx ;
```

【扩展练习】查询每个部门不重复的职位

```
SQL>  select distinct deptno, job from emp_xxx ;
        --distinct 指所有列的唯一组合
```

大小写敏感

【案例 13】职位是 Analyst 的员工数据？

```
SQL>  select * from emp_xxx where job = 'Analyst' ;
                                  --如果数据是 analyst，查不出结果
```

注意：SQL 语句大小写不敏感，数据大小写敏感

【案例 14】查找职位为 "analyst" 的员工

```
SQL>  select * from emp_xxx where lower(job) = 'analyst' ;
```

4.9. upper()函数

将数据转换为大写

【案例 15】忽略大小写，查找职位为 "analyst" 的员工

```
SQL>  select * from emp_xxx where upper(job) = 'ANALYST' ;
```

Between and 的用法

【案例 17】入职时间在 2011 年的员工？

```
SQL>  select * from emp_xx
        where hiredate between '01-JAN-11' and '31-DEC-11' ;
        --闭区间( 包括边界值 )：['01-JAN-11' , '31-DEC-11']
```

In 的用法

4.11. in(列表)

【案例 18】列出职位是 Manager 或者 Analyst 的员工

```
SQL>  select * from emp_xxx
        where job = 'Manager' or job = 'Analyst' ;
```

等同于

```
SQL>  select * from emp_xxx
        where job in ('Manager' , 'Analyst') ;
```

查询数据库中有多少个表

```
select count(*) from user_tables;
```

查询处各个表的名字

```
select table_name from user_tables;
```

" S_" 的模糊查询方法

【案例 22】查询数据库中有多少个名字中以 'S_' 开头的表？

```
SQL>  select count(*) from user_tables
        where table_name like 'S\_%' escape '\' ;

        -- 如果要查询的数据中有特殊字符( 比如_或% ),
        -- 在做模糊查询时，
        -- 需要加上\符号表示转义，并且用 escape 短语指明转义字符\
```

例如

```
select count (*) from user_tables
where table_name like 'his\_' escape '\' ;
```

记住限制条件在 where 的后面!

为空时的写法

■ 正确写法

```
SQL>  select * from emp_xxx where bonus is null ;
```

不为 null 时的写法

【案例 24】哪些员工有奖金？

```
SQL>  select * from emp_xxx where bonus is not null ;
```

Not between and

4.14.2. not between 低值 and 高值

【案例 25】薪水不在 5000 至 8000 的员工？

```
SQL>  select * from emp_xxx where salary not between 5000 and 8000 ;
```

4.14.3. not in (list)：不在列表中

【案例 26】不是部门 20 和部门 30 的员工？

```
SQL>  select * from emp_xxx where depno not in( 20 , 30 ) ;
```

指定字段插入值

【案例 6】插入一条 ID 为 1011，姓名为 '余泽成'，其余字段为 null 的数据

繁琐

```
SQL>  insert into emp_xxx
       values( 1011 , '余泽成' , null , null , null , null , null , null ) ;
```

简写

```
SQL>  insert into emp_xxx( empno , ename )
       values( 1011 , '余泽成' ) ;
```

查看有多少张表?

【案例 22】查询数据库中有多少个名字中以 'S_' 开头的表？

```
SQL> select count(*) from user_tables where table_name like 'S\_%' escape '\';
```

Round 的用法

【案例 1】计算金额的四舍五入

```
SQL>  select ename , salary * 0.1234567  s1 ,            --原样显示
              round( salary * 0.1234567 , 2 )  s2 ,      --保留 2 位有效数字
```

2.1.2. 数字函数 trunc() **

trunc(数字 ， 小数点后的位数)用于**截取**

- ✓ 如果没有第二个参数 ， 默认是 0

Sysdate 的用法

【案例 3】获取系统当前时间

```
SQL>  select sysdate from dual ;                    --dual 为虚表
```

现在的日期减去入职的日期，(sysdate-hiredate) 一共工作了多长时间

- ✓ 日期数据相减，得到两个日期之间的天数差，不足一天用**小数表示。**可以用 round 函数处理一下。

```
SQL>  select ename , hiredate , round( sysdate - hiredate ) days
       from emp ;
```

二、Oracle 各个函数

1 日期函数 months_between()　入职多少个月

【案例 5】计算员工入职多少个月？小数。

```
SQL>  select ename , hiredate ,
                    months_between( sysdate , hiredate ) months
        from emp_xxx ;
```

【案例 6】计算员工入职多少个月？用整数表示

```
SQL>  select ename , hiredate ,
              round( months_between( sysdate , hiredate ) ) months
      from emp_xxx ;
```

2 日期函数 add_months()

【案例 7】计算 12 个月之前的时间点

```
SQL>  select add_months(sysdate, -12) from dual;
```

【案例 4】将 amy 的入职日期提前 2 个月

```
SQL>  select ename , hiredate from emp_xxx
       where ename='amy';
SQL>  update emp_xxx set hiredate=add_months(hiredate  , -2)
       where ename='amy';
SQL>  select ename , hiredate from emp_xxx
       where ename='amy';
```

案例：查找入职已经 1 年的员工

Select ename , hiredate from emp where sysdate>**add_months(hiredate,12);**

3 last_day()

【案例 8】计算本月的最后一天

```
SQL>  select  last_day(sysdate) from dual;
```

案例：查询哪些员工是在月底的倒数第三天入职的！

Select ename , hiredate from emp where **last_day(hiredate)-2**=hiredate;

4 to_char 日期转换函数 to_date

2.3.1. 转换函数 to_char(日期数据 ， 格式): 把日期数据转换为字符数据

【案例 9】把时间数据按指定格式输出

```
SQL>  select  to_char( sysdate ,  'yyyy-mm-dd hh24 : mi : ss ' )
       from dual ;
```

to_date 的用法

【案例 14】按指定时间格式插入数据

```
SQL>  insert  into  emp_xxx( empno , ename , hiredate )
       values( 1012 ,  'amy ' ,
                  to_date(  '2011-10-10 ' ,  'yyyy-mm-dd ' ) ) ;
```

```
select T.TG_MTAL_NAME, sum(t.mesfctrafmval) from VS_MB_MOVEDATA_V t WHERE
to_char(t.begtime,'yyyymmdd')= '20130201'
and t.endtime = TO_DATE ('20130228', 'yyyyMMdd') and t.tg_node_name = '
一催化' group by T.TG_MTAL_NAME;
```

1)　常用日期格式

✓	yyyy	四位数字年　如：2011
✓	year	全拼的年 如：twenty eleven
✓	month	全拼的月 如：november 或 11 月(中文)
✓	mm	两位数字月 如：11
✓	mon	简拼的月 如：nov(中文没有简拼)
✓	dd	两位数字日
✓	day	全拼的星期 如：tuesday
✓	dy	简拼的星期 如：tue
✓	am	上午/下午 如：am/pm

5 coalesce(参数列表)函数的作用。

返回列表中**第一个非空参数**，参数列表中**最后一个值通常为常量**

【案例 16】计算员工的年终奖金

要求：

1) 如果 bonus 不是 null，发年终奖金额为 bonus
2) 如果 bonus 是 null，发年终奖金额为 salary * 0.5
3) 如果 bonus 和 salary 都是 null，发 100 元安慰一下

```
SQL>  select ename , bonus , salary ,
                    coalesce( bonus , salary*0.5 , 100 ) bonus
```

6 case 语句

case 语句斯数据中的分支语句,相当于 Java 中的 swith-case 语句

案例：根据员工的职位，计算加薪后的薪水数据

要求：

1) 如果职位是 Analyst：加薪 10%
2) 如果职位是 Programmer：加薪 5%
3) 如果是 clerk：加薪 2%
4) 其他职位：薪水不变

```
SQL>  select ename , salary , job ,
        case job when  'Analyst'  then  salary * 1.1       --注意这里没有 “ , ”
                 when  'Programmer'  then  salary * 1.05
                 when  'clerk'  then  salary * 1.02
                 else salary          --else 相当于 Java 中 case 语句的 default
        end new_salary                --end 是 case 语句的结束标识
```

7 decode 函数

Decode 函数是 oracle 中等价与 case when 语句的函数，作用 case 语句相同

Decode 函数语法如下

```
decode(判断条件 , 匹配 1 , 值 1 , 匹配 2 , 值 2 , ... , 默认值)
表达的意思是：如果判断条件 = 匹配 1 ,  则返回值 1
              判断条件 = 匹配 2 ,  则返回值 2
```

【案例 18】根据员工的职位 ，计算加薪后的薪水数据

要求：和 case 语句相同

1) 如果职位是 Analyst：加薪 10%
2) 如果职位是 Programmer：加薪 5%
3) 如果职位是 clerk：加薪 2%
4) 其他职位：薪水不变

```
SQL>  select ename , salary , job ,
       decode( job , 'Analyst' , salary * 1.1 ,
                     'Programmer' , salary * 1.05 ,
                     'clerk' , salary * 1.02 ,
                               salary)
       new_salary
       from emp_xxx ;
```

✓ 如果 job 为 Analyst 则 salary*1.1 ;如果为 Programmer 则 salary*1.05 ;如果为 clerk 则 salary*1.02 ;否则值为 salary(不变)

8 排序 order by

【案例 19】薪水由低到高排序(升序排列)

```
SQL>  select ename , salary from emp_xxx
        order by salary asc ;                  --正序排列 , asc 可以省略
```

【案例 20】薪水由高到低排序(降序排列)

```
SQL>  select ename, salary from emp_xxx
        order by salary desc ;                 --desc( descend )降序排列 不可省略
```

多个字段排序

【案例 22】按部门排序 ，同一部门按薪水由高到低排序

```
SQL>  select ename , deptno , salary
       from emp_xxx
       order by deptno , salary desc ;
```

注意：排序语句应该放在查询语句的最后面

9 all

【案例 11】查询谁的薪水比所有叫张无忌的薪水都高？　--大于最大值

```
SQL>  select ename , salary from emp_xxx where ename='张无忌' ;
SQL>  select ename from emp_xxx
       where salary > ALL( select salary from emp_xxx
                           where ename = '张无忌' ) ;
```

10 any

【案例 12】哪些人的薪水比叫张无忌的&&工资最低的人高？　--大于最小值

```
SQL>  select ename from emp_xxx
       where salary > ANY( select salary from emp_xxx
                           where ename = '张无忌' ) ;
                  --只要大于叫 “张无忌” 的人的薪水最小值就查出来
```

11 in

谁和 XXX 在同一部门?列出除他之外的所有的成员!

Select ename ,salary,job,deptno from emp_xx

Where deptno in

(select deptno　from emp_xx where ename=’XXX’)

and ename<>’XXX’

12 exists

【案例 22】哪些人是其他人的经理？(查找有下属的员工)

emp_xxx 表

编码	姓名	职位	薪水	奖金	入职时间	经理	所在
1001	张无忌	Manager	10000	2000	12-MAR-10	1005	10
1002	刘苍松	Analyst	8000	1000	01-APR-11	1001	10
1003	李翔	Analyst	9000	1000	11-APR-10	1001	10
1004	郭芙蓉	Programmer	5000	null	01-JAN-11	1001	10
1005	张三丰	President	15000	null	15-MAY-08	null	20

- 方法 1：使用关联子查询完成

```
SQL>  select ename from emp_xxx a
        where exists (select  1  from emp_xxx
                        where mgr = a.empno) ;
```

✓ exists 关键字判断子查询有没有数据返回，有则为 ture，没有则为 false
exists 不关心子查询的结果，所以子查询中 select 后面写什么都可以
本例中我们写常量“1”

✓ **sql 执行顺序**从主查询开始，把主查询中的 empno 数据传入子查询，作为条件

【案例 23】哪些人不是别人的经理？

- 方法 1：关联子查询

```
SQL>  select ename from emp_xxx a
        where not exists (select 1 from emp_xxx
                            where mgr = a.empno) ;
```

13union 合集

union 会合并重复数据

【案例 25】合集(union)演示

```
SQL>  select ename , salary from emp_xxx
        where deptno = 10
        union
        select ename , salary from emp_xxx
        where salary > 6000 ;
```

Union all 不会合并重复数据

【案例 26】合集(union all)演示

```
SQL>  select ename , salary from emp_xxx
        where deptno = 10
        union all
        select ename , salary from emp_xxx
        where salary > 6000 ;
```

14 intersect 交集

【案例 27】交集(intersect)演示

```
SQL>  select ename , salary from emp_xxx
        where deptno = 10
        intersect
        select ename , salary from emp_xxx
        where salary > 6000 ;
```

15 minus 差集

【案例 28】差集(minus)演示

```
SQL>  select ename , salary from emp_xxx
        where deptno = 10
        minus
        select ename , salary from emp_xxx
        where salary > 6000 ;
```

16join 内连接

【案例 29】列出员工的姓名和所在部门的名字和城市

```
SQL>  select ename , dname , location
        from emp_xxx e join dept_xxx d
        on e.deptno = d.deptno ;
```

【案例 31】列出员工的姓名和他所在部门的名字，把没有部门的员工也查出来

- 方法 1(left outer join)

```
        -- 结果集中包括有部门的员工和没有部门的员工
        -- 驱动表：emp_xxx
        -- 匹配表：dept_xxx
        -- left outer join 以左边的表为驱动表
SQL>  select e.empno , ename , d.deptno , d.dname , d.location
        from emp_xxx e left outer join dept_xxx d
        on e.deptno = d.deptno ;
```

- 方法 2(right outer join)

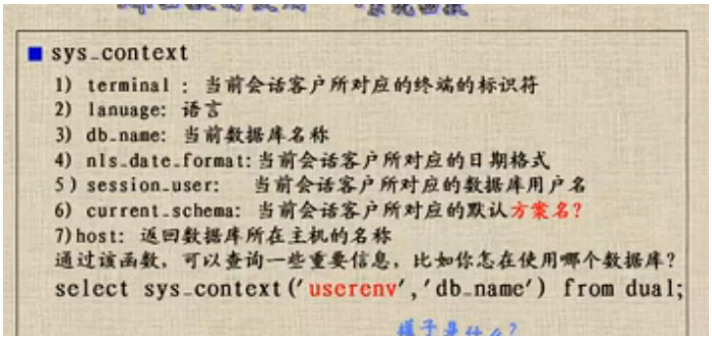
```
        -- 结果集中包括有部门的员工和没有部门的员工
        -- 驱动表：emp_xxx
        -- 匹配表：dept_xxx
        -- right outer join 以右边的表为驱动表
SQL>  select e.empno , ename , d.deptno , d.dname , d.location
        from dept_xxx d right outer join emp_xxx e
        on e.deptno = d.deptno ;
```

【案例 34】全外连接

```
SQL>  select e.empno , e.ename , d.deptno , d.dname
        from emp_xxx e full outer join dept_xxx d
        on e.deptno = d.deptno ;
```

连接类型	定义
内连接	只连接匹配的行
左外连接	包含左边表的全部行（不管右边的表中是否存在与它们匹配的行），以及右边表中全部匹配的行
右外连接	包含右边表的全部行（不管左边的表中是否存在与它们匹配的行），以及左边表中全部匹配的行
全外连接	包含左、右两个表的全部行，不管另外一边的表中是否存在与它们匹配的行。

17 系统函数 sys_context



18 lower 、upper、length、substr、repace

lower(char) 将字符串转化成小写的格式
upper(char)将字符串转化成大写的格式
length（char）：返回字符串的长度；
substr（char，m，n）；取字符串的长度。m 开始 n 表示取几个。
replace（ename,’A’,’******’）;
案例：除首字母后面的都大写
Select upper(substr(ename,2,length(ename)-1)) from emp;

19 round、trunc、floor、ceil

Round（n,[m]）该函数用于四舍五入，去掉 m，则是取整数部分，如果 m 为整数就是取小数点后的 m 位后
Trunc（n，[m]）该函数用于截取数字，去掉 m，则是截取整数部分，如果 m 为整数就是截取小数点后的 m 位
Floor 向下取整 如 55.34　应取 55
Ceil 向上取整 如 55.34 应去 56

17 系统函数 sys_context

18 返回制定字符对应的十进制的数

```
SQL> select ascii('A') A,ascii('a') a,ascii('0') zero,ascii(' ') space from dual;

         A          A       ZERO      SPACE
---------- ---------- ---------- ----------
        65         97         48         32
```

19 给出整数,返回对应的字符

```
SQL> select chr(54740) zhao,chr(65) chr65 from dual;

ZH C
-- -
赵 A
```

20 连接字符串

```
select concat('010-','88888888')||'转 23'　高乾竞电话　from dual
高乾竞电话
----------------
010-88888888 转 23
```

21 next_day(date,’day’);

给出日期 date 和星期 x 之后计算下一个星期的日期
select next_day('18-5 月-2001','星期五') next_day from dual;

```
NEXT_DAY
----------
25-5 月  -01
```

三、 Oracle 组函数

1　count (*)函数的作用

案例：员工列表中有多少条记录
select count(*) from emp_xxx;

2 数据字典　user_tables

user_tables 只读不能改。

【案例 24】当前帐户(openlab)下有多少个表？

```
SQL>   select count(*) from user_tables ;
```

【案例 25】openlab 帐户下有多少个名字中包含 emp 的表？

```
SQL>   select count(*) from user_tables
         where table_name like '%EMP%';
```

查询非空值得数据

【案例 26】入职时间不是 null 的数据总数

```
SQL>   select count(hiredate) from emp_xxx ;    --注意：count 函数忽略空值
```

组函数 count () avg() sum() max() min() **

✓ 薪水平均值 = 薪水总和 / 人数总和　avg(salary) = sum(salary) / count(*)
而 avg(salary)只按有薪水的员工人数计算平均值。这样得到的数据不够准确。

正确写法：

```
SQL>   select count(*) num , sum(salary) sum_sal ,
         avg(nvl(salary , 0)) avg_sal ,
         from emp_xxx ;
```

注意：

✓ 组函数：count / avg / sum / max / min 如果函数中写列名，默认忽略空值
✓ avg / sum 针对数字的操作
✓ max / min 对所有数据类型都可以操作

3 分组函数 group by

```
SQL>   select    deptno , max(salary) max_s , min(salary) min_s
         from emp_xxx
         group by deptno ;
```

【案例 34】按职位分组，每个职位的最高、最低薪水和人数？

```
SQL>   select   job , max(salary) max_s ,
                      min(salary) min_s ,
                      count(*) emp_num
         from emp_xxx
         group by job
         order by emp_num ;
```

注意：select 后出现的列，凡是没有被组函数包围的列，必须出现在 group by 短语中，如上例中的 job

如果 select 后没有被组函数的列，没有出现在 group by 短语中，会出错：

【案例 36】如果 group by 短语中的列，没有出现在 select 短语中，不会出错，信息不够全

```
SQL>   select max(salary) max_s ,
                min(salary) min_s ,
                sum(salary) sum_s ,
                avg( nvl(salary , 0) ) avg_s ,
                count(*) emp_num
         from emp_xxx
         group by deptno ;
```

查询结果中没有部门信息：

MAX_S	MIN_S	SUM_S	AVG_S	EMP_NUM

4 having 子句

Having 子句用于对分组数据进行过滤

注意区别 where 是对表中数据的过滤；having 是对分组得到的结果数据进一步过滤。

【案例 37】平均薪水大于 5000 元的部门数据，没有部门的不算在内？

```
Select deptno.avg(nvl(salary,0)) avg_s from emp_xx
Where deptno is not null
group by deptno
having avg(nvl(salary,0))>5000;
```

【案例 38】薪水总和大于 20000 元的部门数据？

```
SQL>   select deptno , sum(salary) sum_s
         from emp_xxx
         where deptno is not null
         group by deptno
         having sum(salary) > 20000 ;
```

四、MySQL 其他相关知识

1 分类知识点

1.1 更改 mysql 数据库密码（自己设定为 admin）

通过MySQL命令行，可以修改MySQL数据库的密码，下面就为您详细介绍该MySQL命令行，如果感兴趣的话，不妨一看。

格式：mysqladmin -u用户名 -p旧密码 password 新密码

1、给root加个密码ab12。首先在DOS下进入目录mysql\bin，然后键入以下命令

```
mysqladmin -u root -password ab12
```

注：因为开始时root没有密码，所以-p旧密码一项就可以省略了。

2、再将root的密码改为djg345。

```
mysqladmin -u root -p ab12 password djg345
```

（注意：和上面不同，下面的因为是MYSQL环境中的命令，所以后面都带一个分号作为命令结

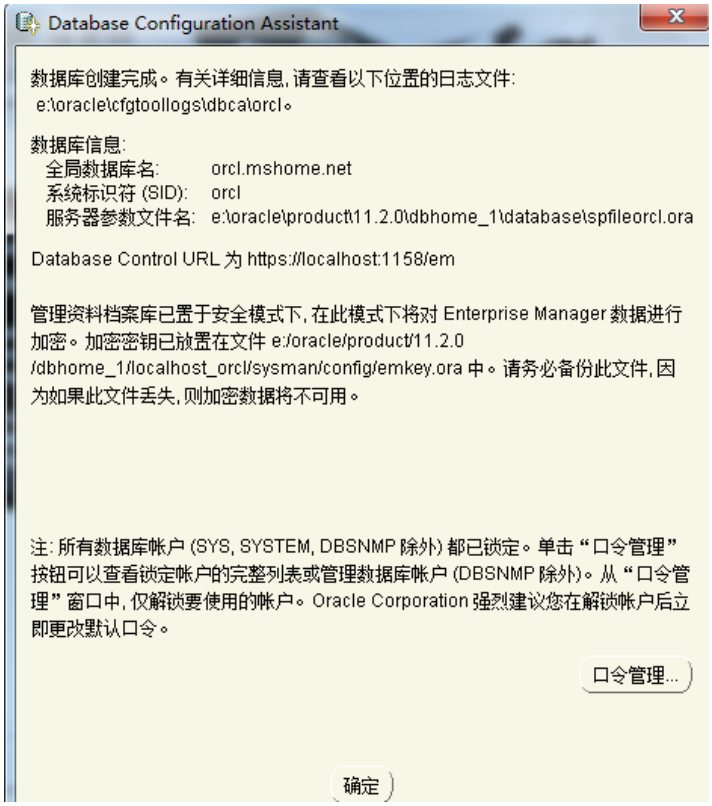
1.2 更改 mysql 数据库密码
1.2 更改 mysql 数据库密码
1.2 更改 mysql 数据库密码
2 分类知识点
2.1 更改 mysql 数据库密码
2.1 更改 mysql 数据库密码

五、Oracle 设置与基础知识点

1 安装图解

口令是：orcl

简单概要：oracle 系统标志符



管理数据库：https://localhost:1158/em

口令管理：

oracle 安装会自动生成 **sys** 用户和 **system** 用户

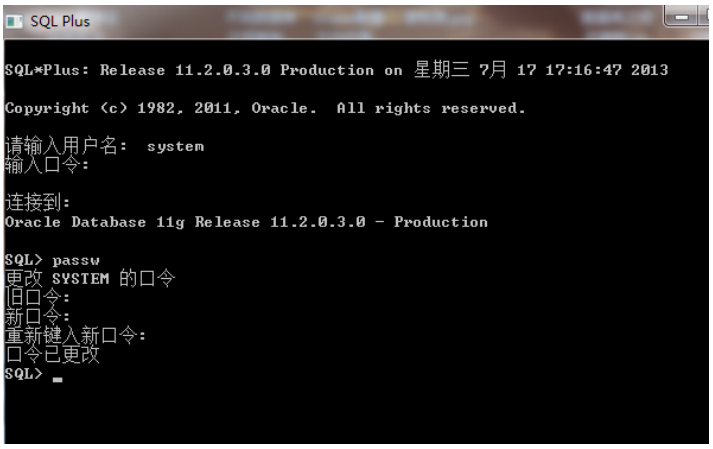
1）sys 用户 是超级用户 ，具有最高权限，具有 sysdba 角色，有 create database 的权限，该用户的的密码是 **manager**

2）system 用户是 管理操作员，权限也很大，具有 sysoper 角色，没有 create database 的权限，默认的密码是 change-on-install

3)一般来讲，对于数据库的维护，使用 system 用户登录就可以了！

2 用户、密码、权限、角色管理

2.1 密码-passw



passw　user（用户名）[只用权限大的才能修改其他用户]

2.2 切换登陆用户

conn　scott/tiger　;　注：此处为反斜线

Scott：默认密码“tiger”，是 oracle 一个普通用户示范

2. 3 修改 oracle 的 scott 用户的密码为默认

用 system 登陆

alter user scott identified by tiger;

alter user scott account unlock （lock 锁定用户）;

2.4 创建一个用户

Create User user Identified By password

user：　用户名称。

password：　用户密码。

2.5 权限的分配

1) Oracle 用户权限分为“系统权限”和“对象权限”2 种。

系统权限：允许用户执行一些数据库操作，如创建表空间等等。

对象权限：允许用户对某一特定对象（如表，视图等）执行特定操作

Grant 命令可以用来分配权限。

分配系统权限：

提供了三种标准的角色(role)：connect、resource 和 dba。

Grant Connect To Fly

Grant Resource To Fly

Grant DBA To Fly

Connect 角色允许用户连接数据库并在数据库创建表或其他对象。

Resource 角色允许用户使用数据库的空间。

DBA 角色允许用户执行数据库操作。

其中 Connect 和 Resource 权限必须分配给用户。

建用户：create user user_name identified by password;

赋权限：grant resource,connect to user_name;

分配对象权限：

select 、 insert、 update、 all 、delete

Grant 对象权限 On 表名称 To 用户

Grant Insert ，select On Emp To Fly　将在 Emp 表中插入、查询数据的权限分配个用户 Fly

Grant All On Emp To Fly　将在 Emp 表中增，删，查，改分配个 Fly。

修改用户密码： Alter User 用户 Identified By 密码

如： Alter User Fly Identified By 333333

删除用户： Drop User Fly

2) 解决 Scott 用户创建视图权限不足问题

Scott 默认是没有创建视图的权限的

我们需要赋予它这个权限

首先用 system 用户登录

conn system/password

然后执行

grant create view to scott ；

提示授权成功

现在 Scott 就拥有创建视图的权限了

3) **希望小明用户可以查询 scott 的 emp 表/还希望小明可以把这个权限继续给别人**。

grant select on emp to xiaoming with grant option。

若果是对象权限就在后面加 with grant option；

（scott 用户自己的表可以授权给别人）

4) 问题：如果 scott 把 xiaoming 对 emp 的权限回收了，那么小红怎么样（被回收）

系统的权限不依赖任何东西，所以级联授权后不级联收回

对象的权限互相依赖，级联授权的后级联的收回

5)如果该用户中没有此表：

select * fromg scott.emp;(**方案：**即使 scott 的表)

6)回收权限

grant all on emp to xiaoming。这张表的所有权限

scott 希望收回 xiaoming 对 emp 的查询权限

revoke select on emp from xiaoming；

7)数据对象

用户创建的表、视图、触发器、存储过程（product）、函数、

8) 查询用户拥有哪里权限

```
select * from dba_role_privs;
select * from dba_sys_privs;
select * from role_sys_privs;
```

3 oracle 数据库相关命令知识点

3.1 显示当前用户　show user　；

3.2 文本操作命令

1> start 和 @

说明：运行 sql 脚本

案例：start e:\qq.sql 或者 @ e:\ss.sql

(在 sqlplus 中运行)

3.3 edit

说明：该命令可以编辑制定的 esql 脚本

案例：edit e:\qq.sql

3.4 spool

说明：该命令可以将 sql*plus 屏幕上的内容输入到制定文件中去

案例：sql>spool e:\aa.sql 并输入 sql>spool off

3.5 设置分页 pagesize

实例说明：Set pagesize 3; 每页显示 3 条记录，可用于打印报表时进行相应的设置

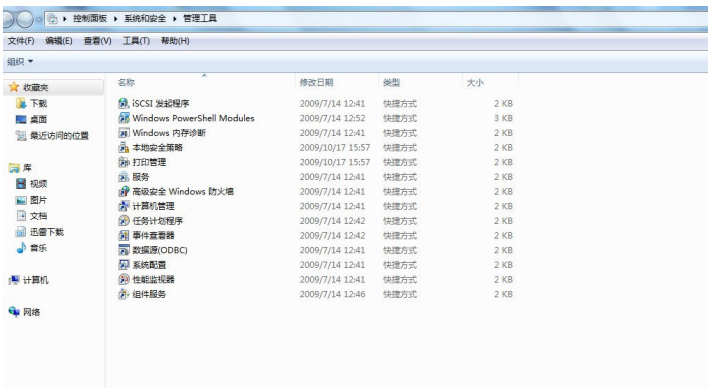
3.6 显示时间：每天操作都显示出操作用的时间

set timing on;

3.7 自身表复制自身表

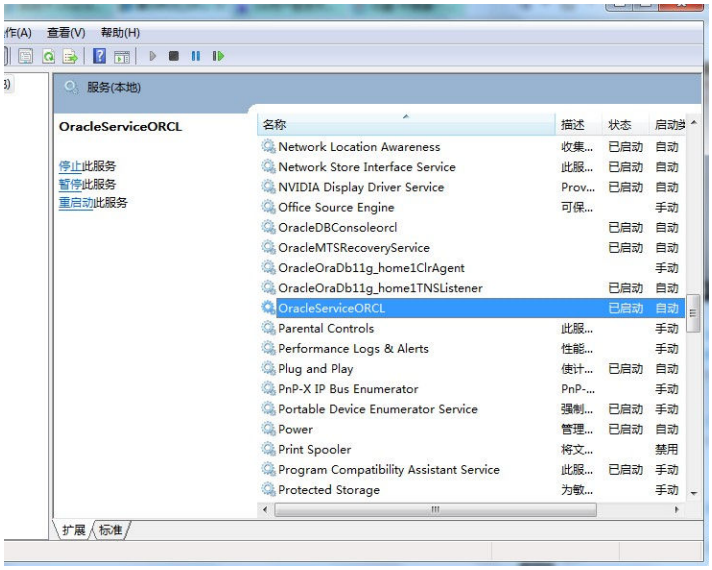
Insert into student（id，name）select * from student；

4 启动 oracle 数据库服务



管理工具--服务

或者 cmd -services.msc



启动 oracle 服务和监听服务（2 个）

启动的是 oracle 单个实例（ORCl），可以创建多个 oracle 数据库实例，那时服务中就会有多个 oracle 实例服务。（可以启动多个实例）

5 配置 oracle 数据库连接 tnsnames.ora

E:\orc\product\11.2.0\dbhome_1\NETWORK\ADMIN\tnsnames.ora

如：mes203 =

(DESCRIPTION =

(ADDRESS_LIST =

(ADDRESS = (PROTOCOL = TCP)(HOST = 10.125.5.203)(PORT = 1521))

)

(CONNECT_DATA =

(SERVICE_NAME = sjzmesdb)

)

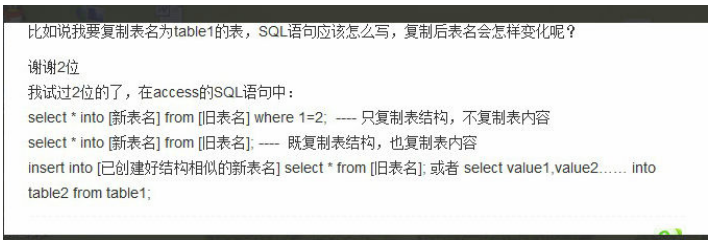
)

6 不同数据之间表的复制

```
create table ys_mes_VS_MB_MOVEDATA as
select * from VS_MB_MOVEDATA_V@mes t
where t.begtime >= TO_DATE ('20121001',
'yyyyMMdd');
```

```
select * from ds_category@ORCL.US.ORACLE.COM
```

注意：不要忘记 **as**



7 卸载 oracle 数据库

8 使用 profile 管理用户口令

概述：profile 是口令限制，资源限制的命令集合，当建立数据库时，oracle

会自动建立名称为 default 的 profile 分配给用户

1）账户锁定

概述：指定该账户登陆时最多可以输入密码的次数，也可以指定用户锁定的时间

的时间（天）一般用 dba 的身份去执行该命令

例如：指定 scott 这个用户最多只能尝试 3 次登陆，锁定时间为 2 天

创建 profile 文件

create　profile lock_account（可改） limit

filed_login_attemps 3 password_lock_time 2;

alter user xiaoming profile lock_account;

给账户解锁

alter user xiaoming　**account unlock；**

删除 profile

概述：当不需要某个 profile 文件时，可以删除该文件

Drop profile password_history [cascade]

六 Oracle 表以及相关查询

1 表的相关知识点

1 建立一个表

学生表的建立：

Create table student(

Xh number(4),　--学号

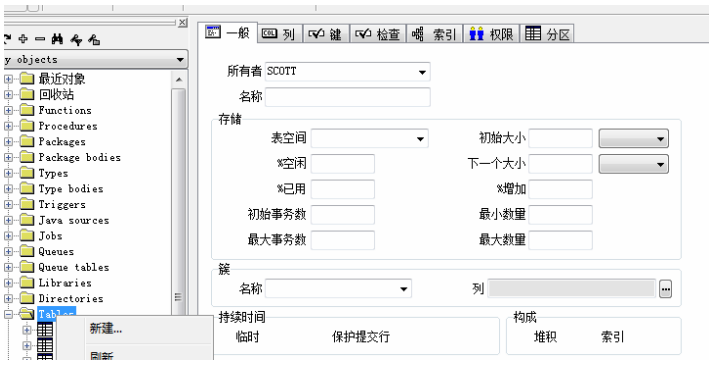
Xm varchar2(20)，--姓名

Birthday date，　--出生日期

Sal number(7,2)　--奖学金

);

PL/SQL 创建表



可以填写表的名称选择 表空间同事可以设置主键、权限等

Clear 清屏命令

2 修改表的结构

1）给表添加一个字段性别字段

Alter table student add(gender char(2))；

2）修改字段的长度

Alter table student　midify(xm varchar2(8));

3）修改字段的类型/或者名字（不能有数据）

Alter table student　midify(xm char(8));

4）删除一个字段

Alter table student drop column xm;

5）修改表的名字

Rename student to stu；

6）删除表

Drop table student；

Delete from student;

Truncate table student; 表结构还在，删除速度快

7）更新表 upadte

Update student set sal=12 ，id=’1’where sex=’男’；

8)回滚命令

Savepoint aa ;

Rollback to aa ;

3 相关查询

1）取消重复行 distinct

Select distinct id ，name from student ;

2）查询数据库中的表

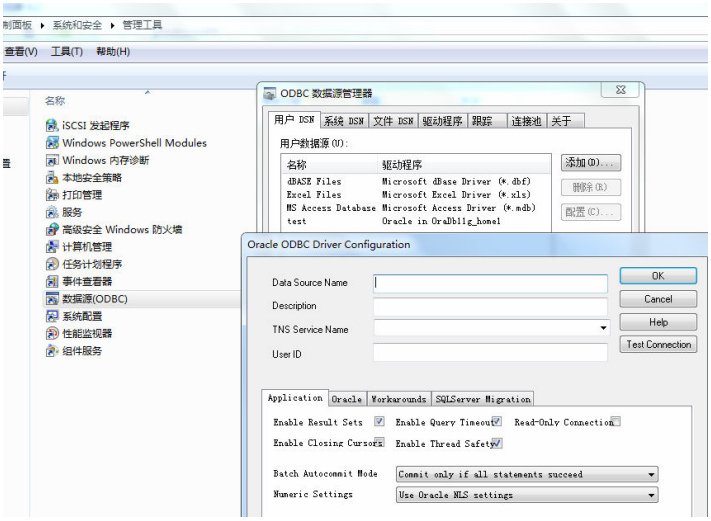
Select table_name from user_tables;显示**当前用户**所有的**表**

(All_tables/dba_tables)

3）查询数据库中的角色

Select * from dba_roles; system 登陆

4 ODBC 连接 oracle 数据库



选好数据库后，可以测试连接

七 PL/SQL 编程

理解 oracle pl/sql 概念 ||掌握 pl/sql 编程技术（包括编写过程，函数，触发器。。。）

它是 oracle 在标准 sql 语言上的扩展，不仅允许嵌入 sql 语句，还可以定义变量和常量，允许使用条件语句和循环语句，允许使用例外处理各种错误，使功能变得更强大。

1 pl/sql 基础 存储过程

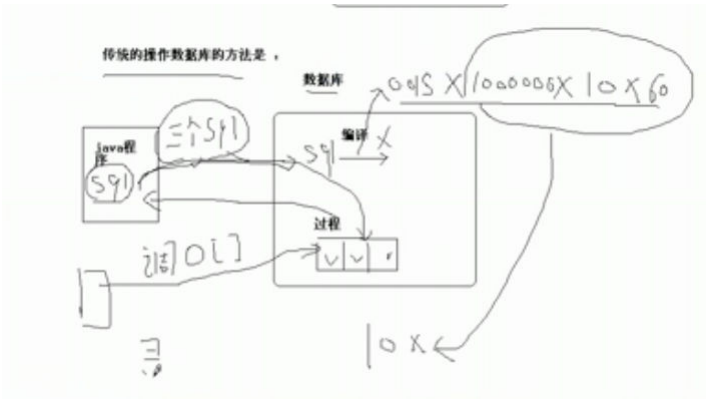
1）学习必要性

(1)提高应用的程序系能

(2)模块化的设计思想【分页过程、订单过程。。。】

(3)减少网络的传输量

(4)提高安全性



缺点：移植性不好【数据库一换，写的就不能用了】

2）案例创建 procedure

（1）案例

```
create or replace procedure  test is
begin
--执行过程
insert into emp_1 values(1,'zx');
end ;
/  输入完后要输入斜杆次才能执行
```

（2）块的基础程序

```
set serveroutput on; 打开输出窗口
--最简单的块
begin
dbms_output.put_line('shu chu hello);
end;
```

（3）块案例：

```
declare
--定义变量
 v_ename varchar2(5);
 v_sal number(7,2);
begin
--执行部分
select ename,sal into v_ename,v_sal from emp where empno=&aa;
```

--在控制台显示用户名

```
dbms_output.put_line('用户名'||v_ename||'工资：'||v_sal);
```

--异常的处理

```
exception
when  no_data_found then
dbms_output.put_line('朋友，你输得编号有误');
end;
```

（4）创建存储过程

```
create procedure test_pro(spName varchar,newSal number) is
begin
--执行部分，根据用户名去修改工资
update emp set sal=newSal where ename=spName;
end;
/
exec test_pro(‘SCOTT’,3000);--执行存储函数（大小写区分）
```

（5）java 程序调用存储过程

```
CallableStatement cs=conn.prepareCall(“{call test pro(?,?)}”);
```

3）基础用到的命令

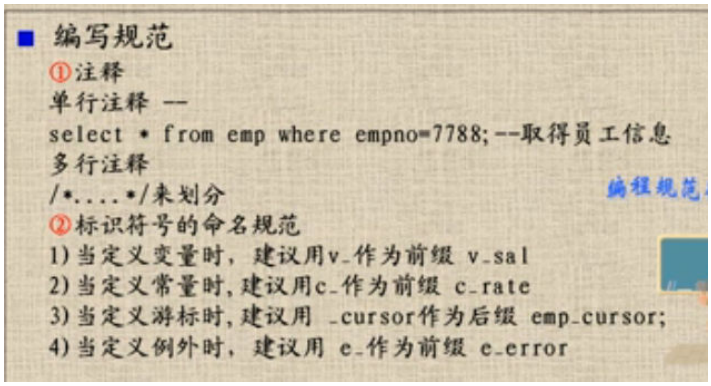
（1）show error;//查看创建过程中出现的错误

（2）如何调用创建的过程

exec 过程名（参数 1，参数 2……）

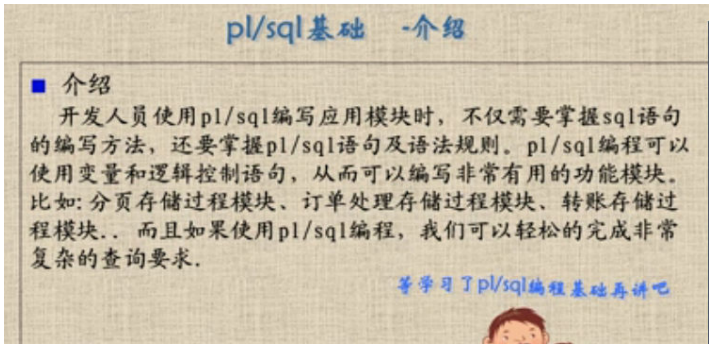
call 过程名（参数 1，参数 2……）

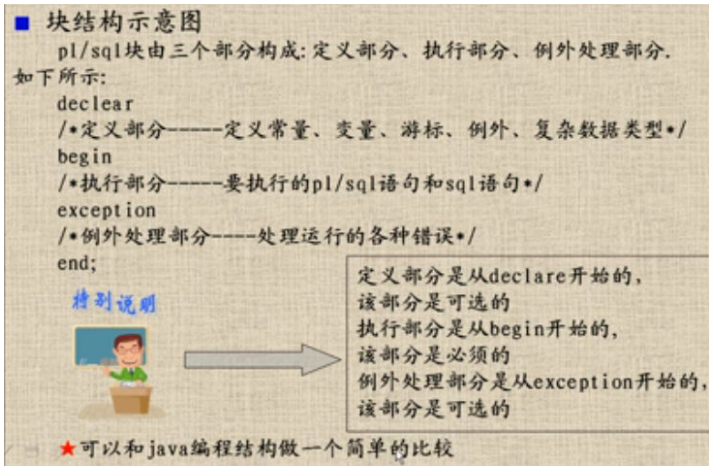
4）编写规范



5）块的概念

块分为（过程、函数、触发器、包）





2　pl/sql 函数、包、触发器

1）案例

```
create function test_fun(name varchar2) return number is yearSal number(7,2);
begin
--执行部分
select sal*12+nvl(comm,0) into yearSal from emp where ename=name;
return yearSal;
end;
```

2）调用函数

在 pl/sql 中调用 ：

var abc number;

call test_fun（‘SCOTT’）into:abc

在 java 程序中调用：

```
ResultSet rs=sm.executeQuery("select test_fun('SCOTT') from dual");
 rs.getInt(1);通过这种方式得到，返回的数据是 int 类型的，yearSal
```

3）创建包

--创建一个包

create package test_pk is

procedure update_sal(name varchar2,newsal number);

function annual_income(name varchar2) return number;

end;

--创建包体

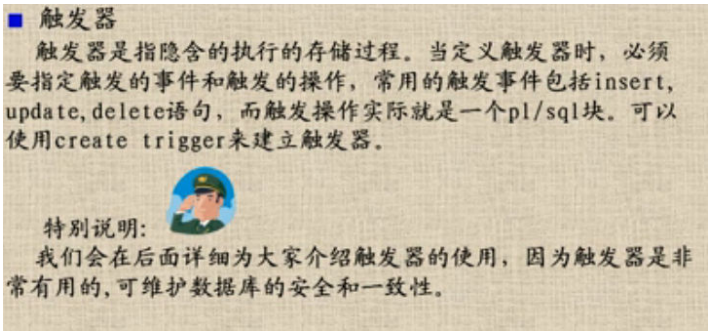
```
②建立包体可以使用create package body 命令
  create package body sp_package is
  procedure update_sal(name varchar2,newsal number)
  is
  begin
    update emp set sal=newsal where ename=name;
  end;
  function annual_income(name varchar2)
  return number is
  annual_salary number;
  begin
    select sal*12+nvl(comm,0) into annual_salary from emp
  where ename=name;
  return annual_salary;
  end;
  end;
```

调用包中的函数

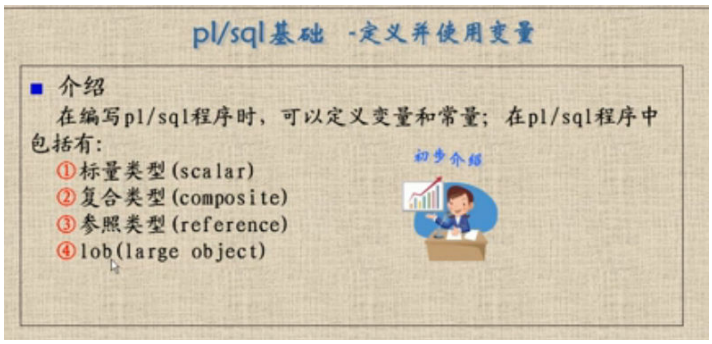
Call sp-package.update_sal（‘SCOTT’,120);

4）触发器

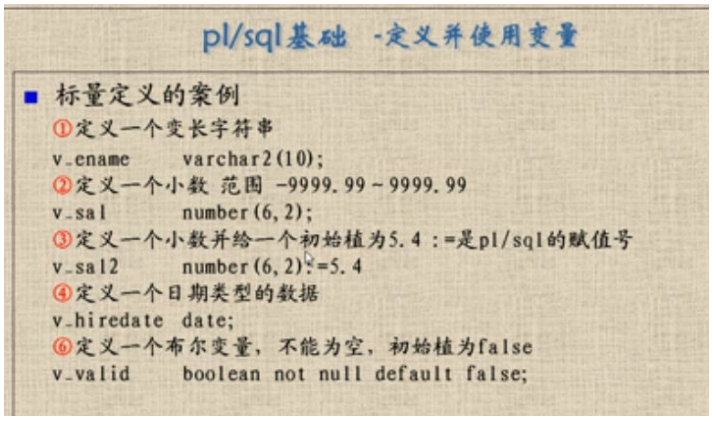
(1)基础知识

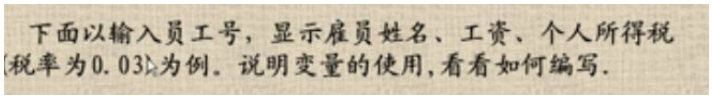


5）定义并使用变量



(1)标量类型





注意：set serveroutput on; 打开输出窗口

```
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
--执行
select ename,sal into v_ename,v_sal from emp where empno=&no;
--计算所得税
v_tax_sal:=v_sal*c_tax_rate;
--输出
dbms_output.put_line('姓名是:'||v_ename||'工资：'||v_sal||'交税'||v_tax_sal);
end;
```

■ 标量(scalar)-使用%type类型

对于上面的pl/sql块有一个问题:

就是如果员工的姓名超过了5字符的话，就会有错误，为了降低pl/sql程序的维护工作量，可以使用%type属性定义变量，这样它会按照数据库列来确定你定义的变量的类型和长度，

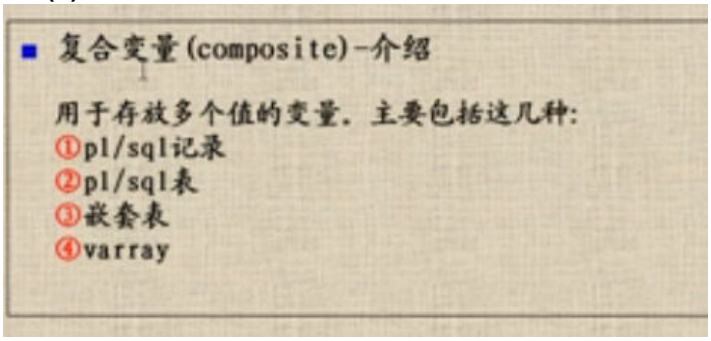
我们看看这个怎么使用:

标识符名 表名.列名%type;

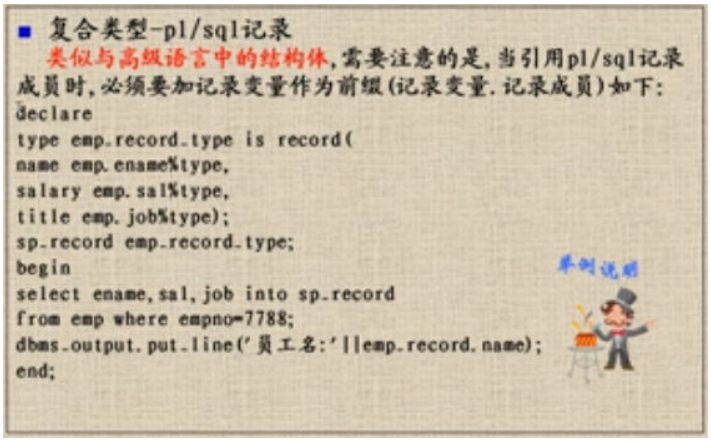
把 v_ename varchar2(5);改为

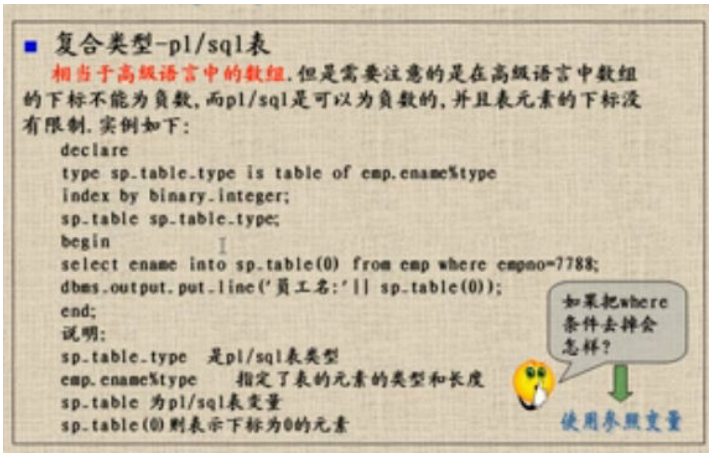
v_ename emp.ename%type

(2)复合变量

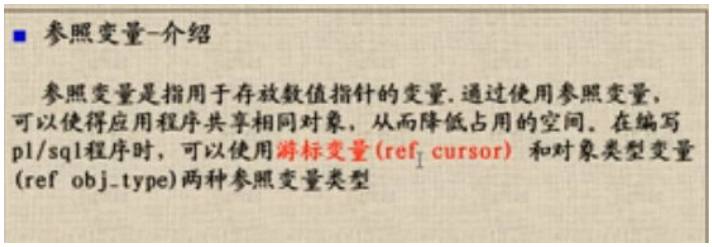


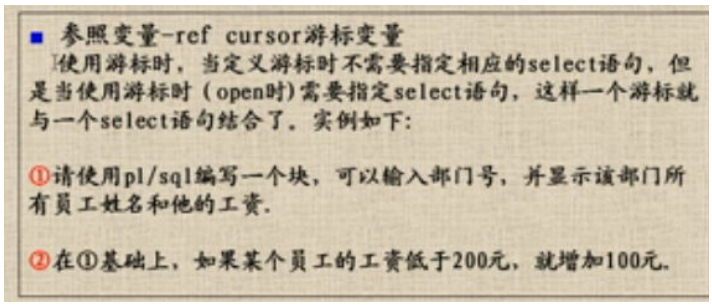
定义 pl/sql 记录类型 emp_record_type,类型包含三个数据 name , salary , title

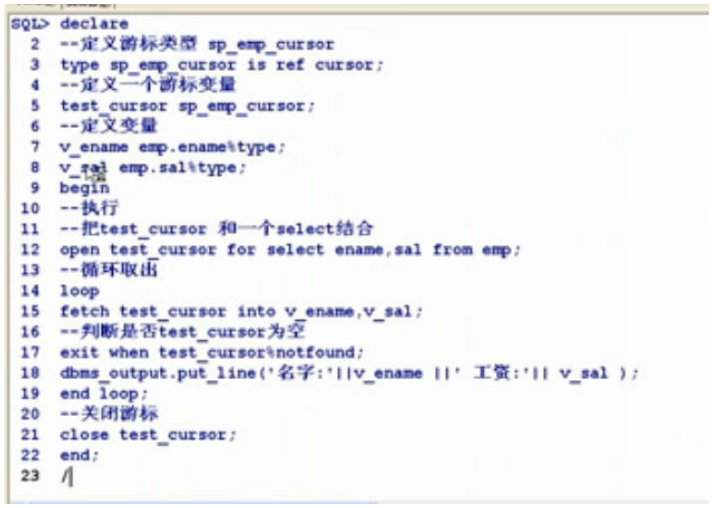




(3)参照类型





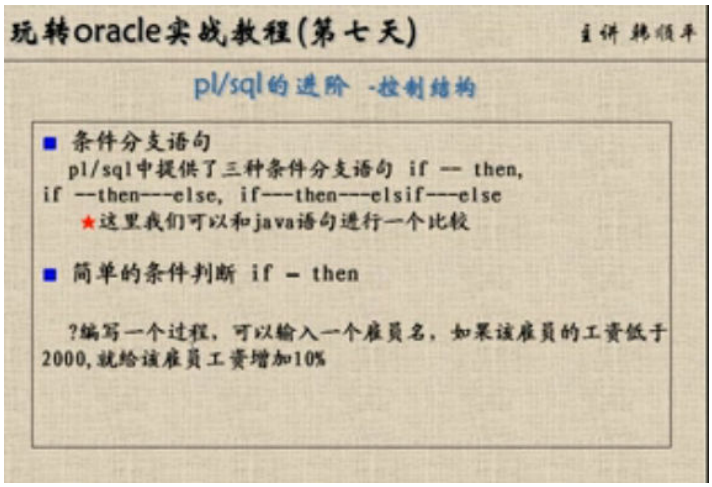


```
SQL> declare
  2  --定义游标类型 sp_emp_cursor
  3  type sp_emp_cursor is ref cursor;
  4  --定义一个游标变量
  5  test_cursor sp_emp_cursor;
  6  --定义变量
  7  v_ename emp.ename%type;
  8  v_sal emp.sal%type;
  9  begin
 10  --执行
 11  --把test_cursor 和一个select结合
 12  open test_cursor for select ename,sal from emp;
 13  --循环取出
 14  loop
 15  fetch test_cursor into v_ename,v_sal;
 16  --判断是否test_cursor为空
 17  exit when test_cursor%notfound;
 18  dbms_output.put_line('名字:'||v_ename ||' 工资:'|| v_sal );
 19  end loop;
 20  --关闭游标
 21  close test_cursor;
 22  end;
 23  /
```

6）控制结构

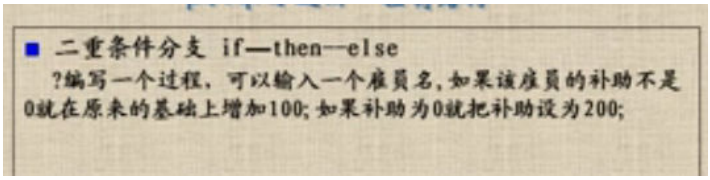
(1) if 选择结构

案例一　if-then



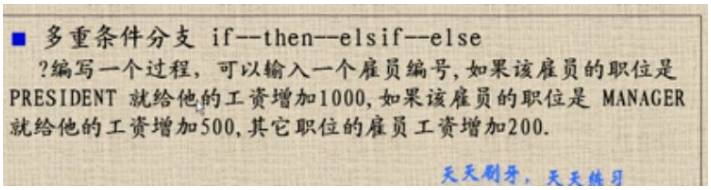
```
SQL>
SQL> create or replace procedure sp_pro6(spName varchar2) is
  2  --定义
  3  v_sal emp.sal%type;
  4  begin
  5  --执行
  6  select sal into v_sal from emp where ename=spName;
  7  --判断
  8  if v_sal<2000 then
  9  update emp set sal=sal*1.1 where ename=spName;
 10  end if;
 11  end;
 12  /
```

案例 2　if-then-else




```
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;
```

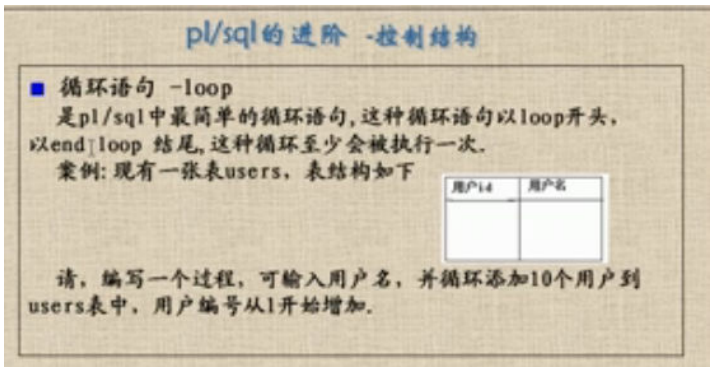
案例三




```
QL> create or replace procedure sp_pro6(spNo number) is
  2  --定义
  3  v_job emp.job%type;
  4  begin
  5  --执行
  6  select job into v_job from emp where empno=spNo;
  7  if v_job='PREDIENT' then
  8  update emp set sal=sal+1000 where empno=spNo;
  9  elsif v_job='MANAGER' then
 10  update emp set sal=sal+500 where empno=spNo;
 11  else
 12  update emp set sal=sal+200 where empno=spNo;
 13  end if;
 14  end;
 15  /

过程被创建
```

上图 presdient 写错，改正确

(2)循环结构



类似 do--while

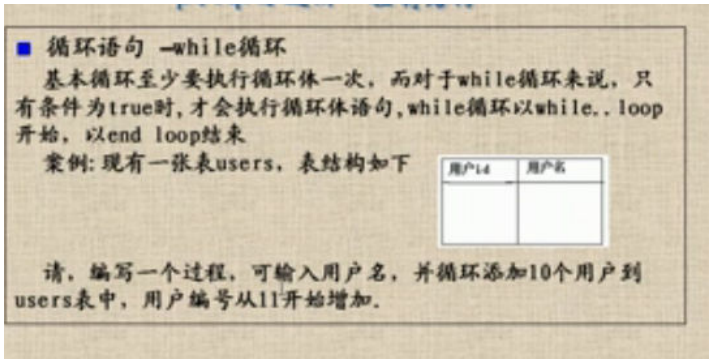
```
> create or replace procedure sp_pro6(spName varchar2) is
  --定义 :=表示赋值
  v_num number:=1;
  begin
  loop
       insert into users1 values(v_num,spName);
       --判断是否要退出循环
       exit when v_num=10;
       --自增
       v_num:=v_num+1;
  end loop;
  end;
  /

程被创建
```

先建立 user1 这张表

```
Name     Type         Nullable Default Comments
-------- ------------ -------- ------- --------
USERNO   NUMBER       Y
USERNAME VARCHAR2(40) Y
```

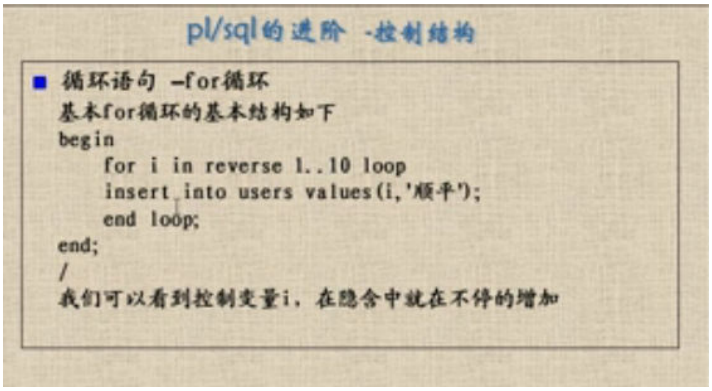
```
SQL> exec sp_pro6('你好');

PL/SQL 过程被成功完成
```

While 循环



```
create or replace procedure sp_pro6(spName varchar2) is
--定义 :=表示赋值
v_num number:=11;
begin
while v_num<=20 loop
--执行
    insert into users1 values(v_num,spName);
    v_num:=v_num+1;
end loop;
end;
```

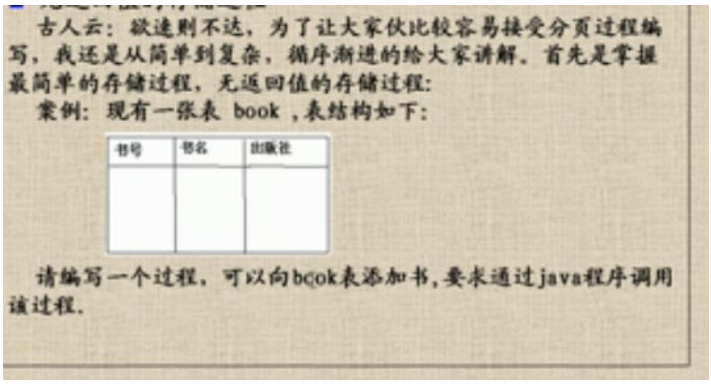
For循环，不太好



Goto 语句，不建议使用破坏了顺序执行的流程

```
SQL>
SQL> declare
  2   i int :=1;
  3   begin
  4     loop
  5     dbms_output.put_line('输出i='||i);
  6     if i=10 then
  7     goto end_loop;
  8     end if;
  9     i:=i+1;
 10     end loop;
 11   <<end_loop>>
 12   dbms_output.put_line('循环结束');
 13   end;
 14  /
```

(3)分页

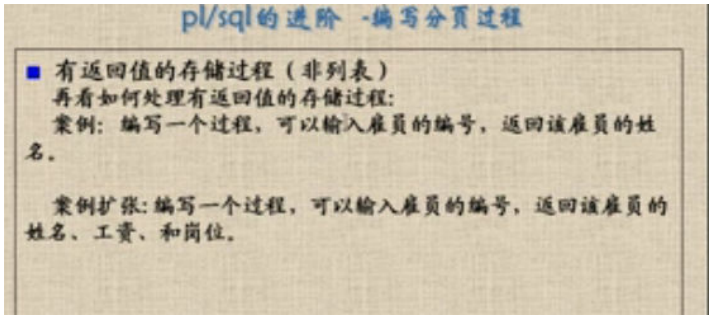


建立 book 表

```
SQL>
SQL> create table book
  2  (bookId number,bookName varchar2(50),publishHouse varchar2(50))
  3  ;

表被创建
```

```
--编写过程
--in: 表示这是一个输入参数,默认为in
--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;
```

案例输入输出



--有输入输出的存储过程（默认省略 in）

create or replace procedure sp_pro8

(spno **in** number,spName **out** varchar2)is

begin

select ename into spName from emp where empno=spno;

end;

Java 调用

```
    //创建CallableStatement
    CallableStatement cs=ct.prepareCall("{call sp_pro8(?,?)}");

    //给第一个?赋值
    cs.setInt(1, 7788);
    //给第二个?赋值
    cs.registerOutParameter(2, oracle.jdbc.OracleTypes.VARCHAR);

    //执行
    cs.execute();
    //取出返回值,要注意?顺序

    String name=cs.getString(2);

    s
    //关闭
```

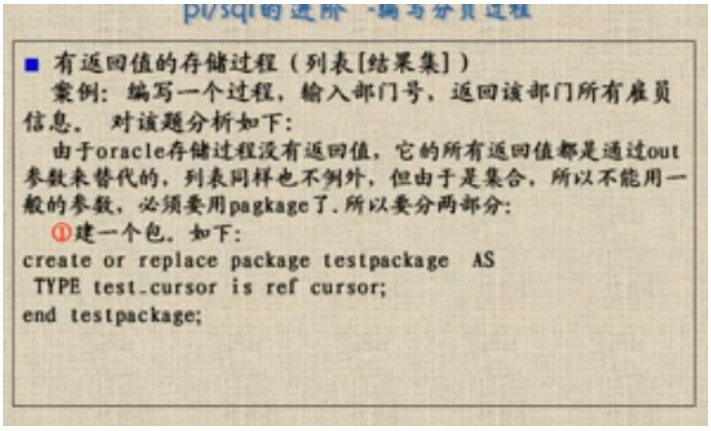
扩展

```
--有输入和输出的存储过程

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;
```

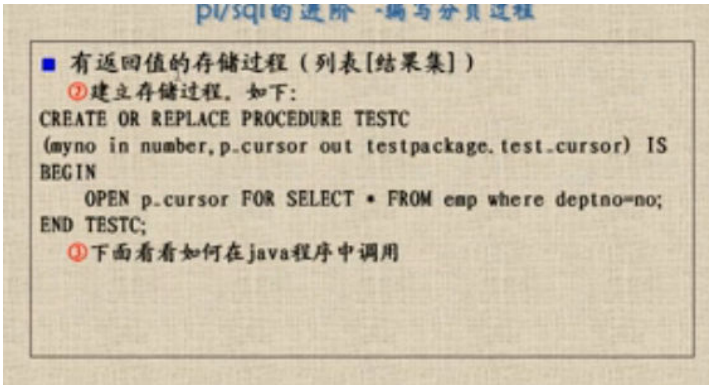
Java 程序必须都得关联

```
//创建CallableStatement
CallableStatement cs=ct.prepareCall("{call sp_pro8(?,?,?,?)}"

//给第一个?赋值
cs.setInt(1, 7788);
//给第二个?赋值
cs.registerOutParameter(2, oracle.jdbc.OracleTypes.VARCHAR);

cs.registerOutParameter(3, oracle.jdbc.OracleTypes.DOUBLE);
cs.registerOutParameter(4, oracle.jdbc.OracleTypes.VARCHAR);
//执行
cs.execute();
//取出返回值,要注意?顺序

String name=cs.getString(2);
```

分页列表



第二步



包是一个比较重要的概念

--返回结果集的过程

--1 创建一个包，在改包中，我定义类型 **test_cursor**,是个游标

create or replace package tespackage as

type test_cursor is ref cursor;

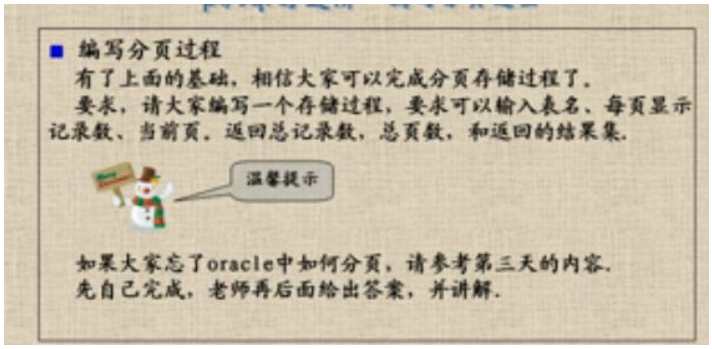
end tespackage;

```
--2.创建过程
create or replace procedure sp_pro9
(spNo in number,p_cursor out tespackage.test_cursor) is
begin
open p_cursor for select * from emp where deptno=spNo;
end;
```

Java 调用

```
//1.创建CallableStatement
CallableStatement cs=ct.prepareCall("{call sp_pro9(?,?)}");

//给?赋值

cs.setInt(1, 10);

cs.registerOutParameter(2, oracle.jdbc.OracleTypes.CURSOR);

//执行
cs.execute();


//得到结果集

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

while(rs.next()){

    sys
}
```

分页




```
--oracle的分页

select t1.*,rownum rn from (select * from emp) t1


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

--在分页时，大家可以把下面的sql语句当做一个模板使用
select * from
(select t1.*,rownum rn from (select * from emp) t1 where rownum<=10)
where rn>=6;


--开始编写分页的过程
```

```
--开发一个包

--1创建一个包,在该包中，我定义类型 test_cursor ,是个游标
create or replace package tespackage as
type test_cursor is ref cursor;
end tespackage;

--开始编写分页的过程
```

```
create or replace procedure fenye
(tableName in varchar2,
Pagesize in number, --一页显示记录数
pageNow in number,
myrows out number,--总记录数
myPageCount out number,--总页数
p_cursor out tespackage.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
||') t1 where rownum<='||v_end||') where rn>='||v_begin;
--把游标和sql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;
```

：=是 赋值的，注意上面

```
CallableStatement cs=ct.prepareCall("{call fenye(?,?,?,?,?,?)

//给?赋值
cs.setString(1, "emp");
cs.setInt(2, 5);
cs.setInt(3, 1);

//注册总记录数
cs.registerOutParameter(4, oracle.jdbc.OracleTypes.INTEGER);
//注册总页数
cs.registerOutParameter(5, oracle.jdbc.OracleTypes.INTEGER);
//注册返回的结果集
cs.registerOutParameter(6, oracle.jdbc.OracleTypes.CURSOR);

cs.execute();
//取出总记录数/这里要注意，getInt(4) 中4，是由该参数的位置决定的

int rowNum=cs.getInt(4);
```

7）Exception 异常

1

■ 预定义例外 no_data_found

下面是一个pl/sql块，当执行select into 没有返回行，就会触发该例外

```
declare
        v_sal emp.sal%type;
begin
        select sal into v_sal from emp
        where ename='&name';
exception
        when no_data_found then
        dbms_output.put_line('不存在该员工');
end;
```

2

■ 预定义例外 case_not_found

在开发pl/sql块中编写case语句时，如果在when子句中没有包含必须的条件分支，就会触发case_not_found的例外：

```
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'||v_sal||'相匹配的条件');
end;
```

3

■ 预定义例外 dup_val_on_index

在唯一索引所对应的列上插入重复的值时，会隐含的触发例外dup_val_on_index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;
```

4

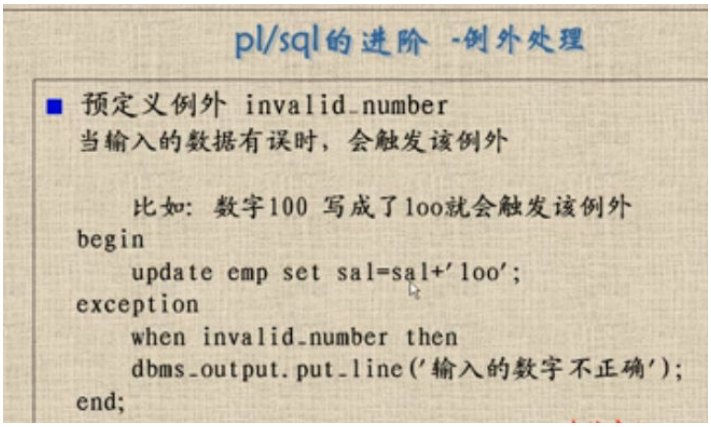
■ 预定义例外 invaild_cursor

当试图在不合法的游标上执行操作时，会触发该例外

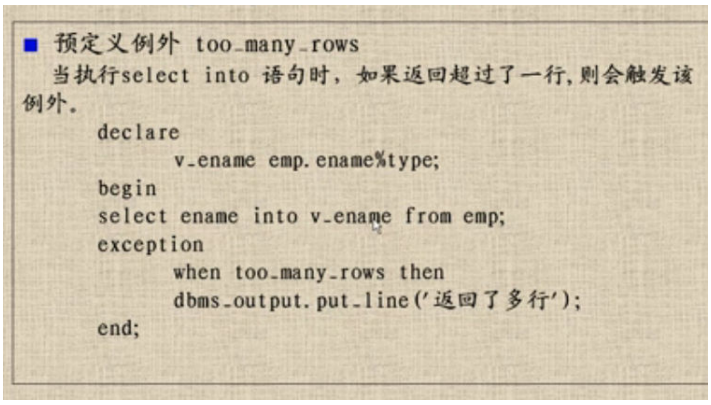
例如：试图从没有打开的游标提取数据，或是关闭没有打开的游标。则会触发该例外

```
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('请检测游标是否打开');
end;
```

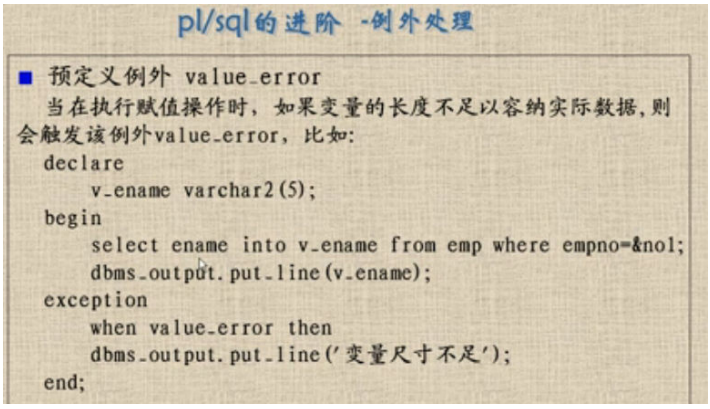
5



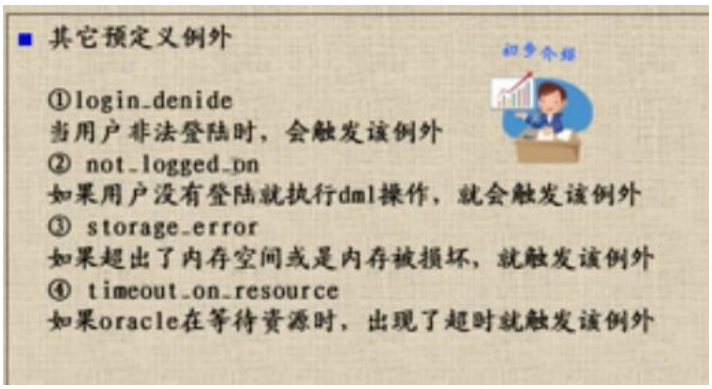
6



7



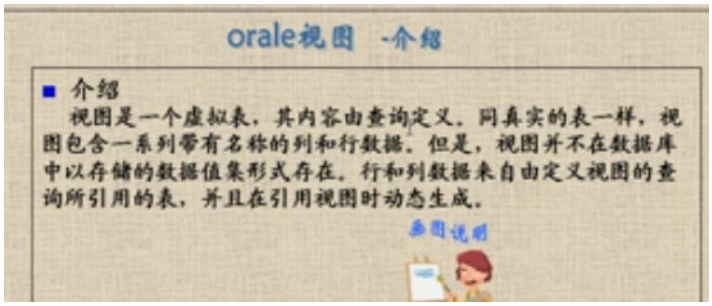
8

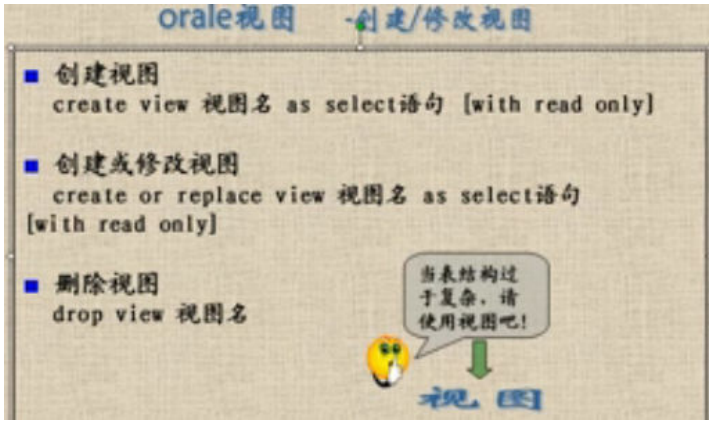


自定义例外

```
L> create or replace procedure ex_test(spNo number)
2    is
3    --定义一个例外
4    myex exception;
5    begin
6    --更新用户sal
7    update emp set sal=sal+1000 where empno=spNo;
8    --sql%notfound这是表示没有update
9    --raise myex;触发myex
0    if sql%notfound then
1    raise myex;
2    end if;
3    exception
4    when myex then
5    dbms_output.put_line('没有更新任何用户');
6    end;
7  /
```

8）视图





第一标题

二标题

1.1 三标题

1.1.1 四标题

1.1.2 四标题

MySql 知识点

第一标题